

Design • Build • Program • Learn • Compete

SERVO

FOR THE ROBOT INNOVATOR

www.servomagazine.com

MAGAZINE

June 2011

BUILDING BOTS FROM FOUND PARTS

Cross-Link
Control System
From Cross
The Road
Electronics

Make your own
"NO-CUT"
Mini T-Bot

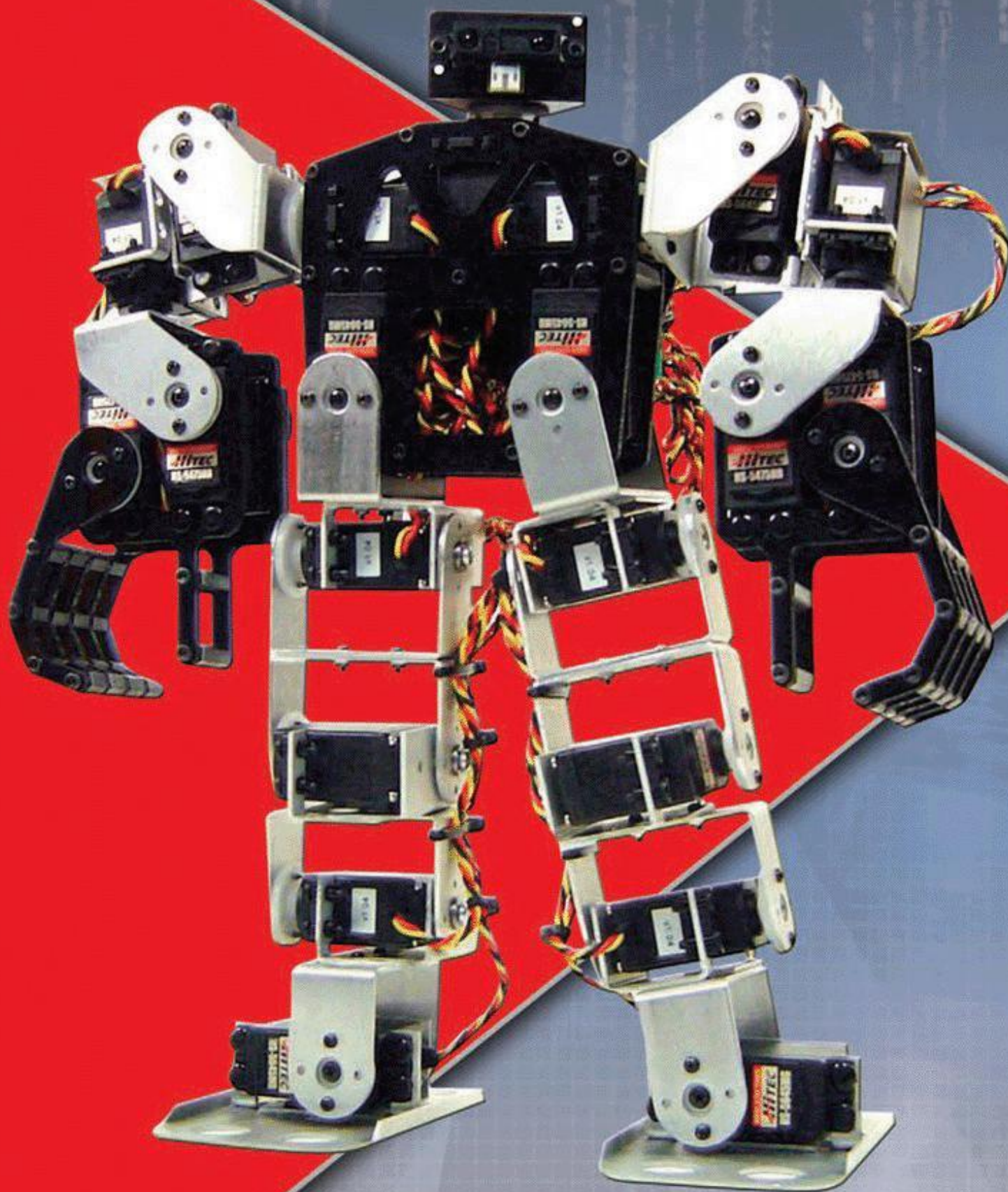
- ♦ New Approaches To Robotics Education
- ♦ Lose The Wires
- ♦ An Alternative Approach To Programming LEGO NXTs

U.S. \$5.50 CANADA \$7.00



GET WITH THE PROGRAM!

Hitec's **HPP-21** and **HPP-21 Plus** are the perfect answer to programming your Hitec digital servos. Whether your robot uses our conventional **HS-5XXX/ HS-6XXX** servo series or the high voltage, ultra premium **HS-79XX** series, our programmers will dependably test and program all your digital servo parameters.



HPP-21
PC Programmer
for Hitec Digital Servos



HPP-21 Plus
PC and Field Programmer
for Hitec Digital Servos

Both programmers will test servos of any brand and offer a USB-PC connection with free downloadable software.





do more wire less

Wixel



actual size

USB. Wireless. Programmable. All for \$19.95.

Introducing the Wixel, a programmable microcontroller module with integrated USB and a 2.4 GHz radio. Write your own program or load pre-compiled, open-source apps to give your next project a wireless serial link, create a remote sensor network, and so much more.



 **Pololu**
Engage Your Brain

learn more at www.pololu.com/wixel

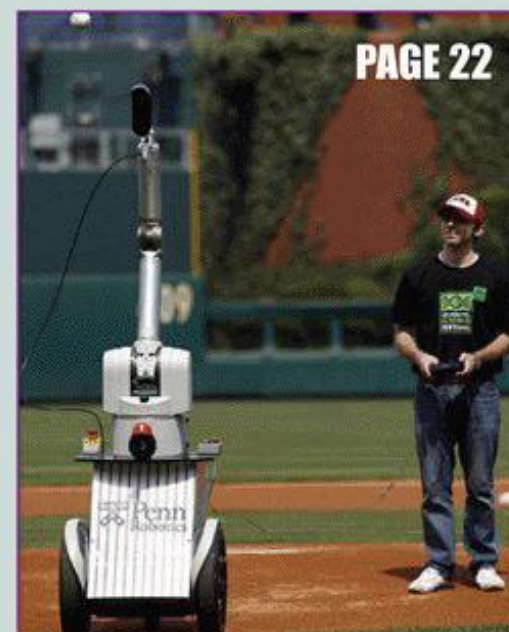
Did you know that each article in *SERVO Magazine* has its own webpage? It's where you go for downloads, comments, updates, corrections, or to link to the article in the digital issue. The unique link for each webpage is included with the article.



PAGE 70

Departments

- 06 Mind/Iron
- 18 Events Calendar
- 20 New Products
- 21 Showcase
- 22 Bots in Brief
- 67 SERVO Webstore
- 81 Robo-Links
- 81 Advertiser's Index



PAGE 22

Columns

- 08 **Robytes**
by Jeff Eckert
Stimulating Robot Tidbits
- 10 **GeerHead**
by David Geer
A Robot's Touch
- 14 **Ask Mr. Roboto**
by Dennis Clark
Your Problems Solved Here

- 62 **The NXT Big Thing #9**
by Greg Intermaggio
In Control!
- 70 **Twin Tweaks**
by Bryce and Evan Woolley
Why Did The Robot Cross The Road?
- 76 **Then and Now**
by Tom Carroll
New Approaches to Robotics Education

The Combat Zone...

Features

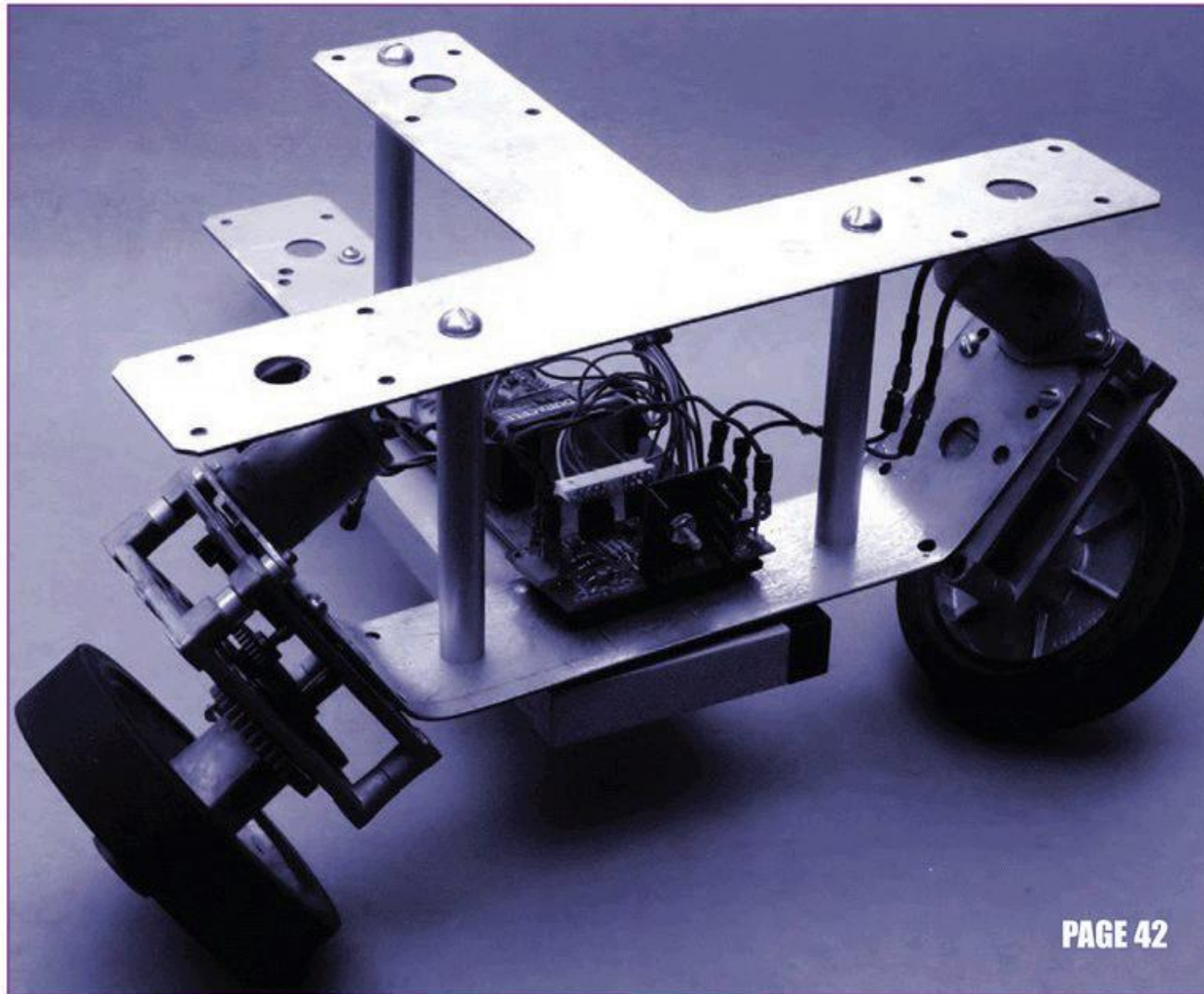
- 26 **BUILD REPORT:**
Moros — A 30 lb Angled Bar Spinner
- 28 **POTPOURRI 3.0**
- 32 **The Reality of TV**

Events

- 32 **Completed and Scheduled Events**



PAGE 26



36 A UART/SPI Monitor for Your Micro — Part 2

by Mark Mitchell

This month, we'll finish off the tool that lets us "peek under the hood" of our controller and then we'll put it into a useable housing.

54 Make Your Robot's Wires Extinct

by Fred Eady

Find out about a new 8051-based powerhouse that will allow you to lose the wire connection to the microcontroller in your robot.

42 Building Bots From Found Parts

by Gordon McComb

There's virtually no limit to the number and type of items you can use in your robot projects. Find out just how easy it is to build your own "no-cut" robot out of ready-made parts available at your local hardware store.

60 Programming the LEGO NXT: An Alternative Approach Suitable For Developing Tomorrow's Engineers

by John Blankenship and Samuel Mishal

LEGO makes building robots easier, and now programming a LEGO NXT can be just as easy. Learn about the open-source LegoLibrary.bas.

49 CPLDs — Part 4: HDL Programming

by David Ward

Now that you have a better idea of what a CPLD is and what it can do, it's time to introduce and begin using HDL (hardware description language).



Mind / Iron

by Bryan Bergeron, Editor



Embedded Linux Development Platform

I had the pleasure of speaking with a fellow robotics enthusiast, Eric Gregori, about his latest contribution to the field of affordable robotics. Eric started building robots at eight, and continues his passion/obsession today as the Embedded Firmware Product Specialist at Freescale Semiconductors. Because Freescale is into selling silicon and Eric is into robotics, it seems as though he's one of the lucky few people that get paid for what they would otherwise do for free.

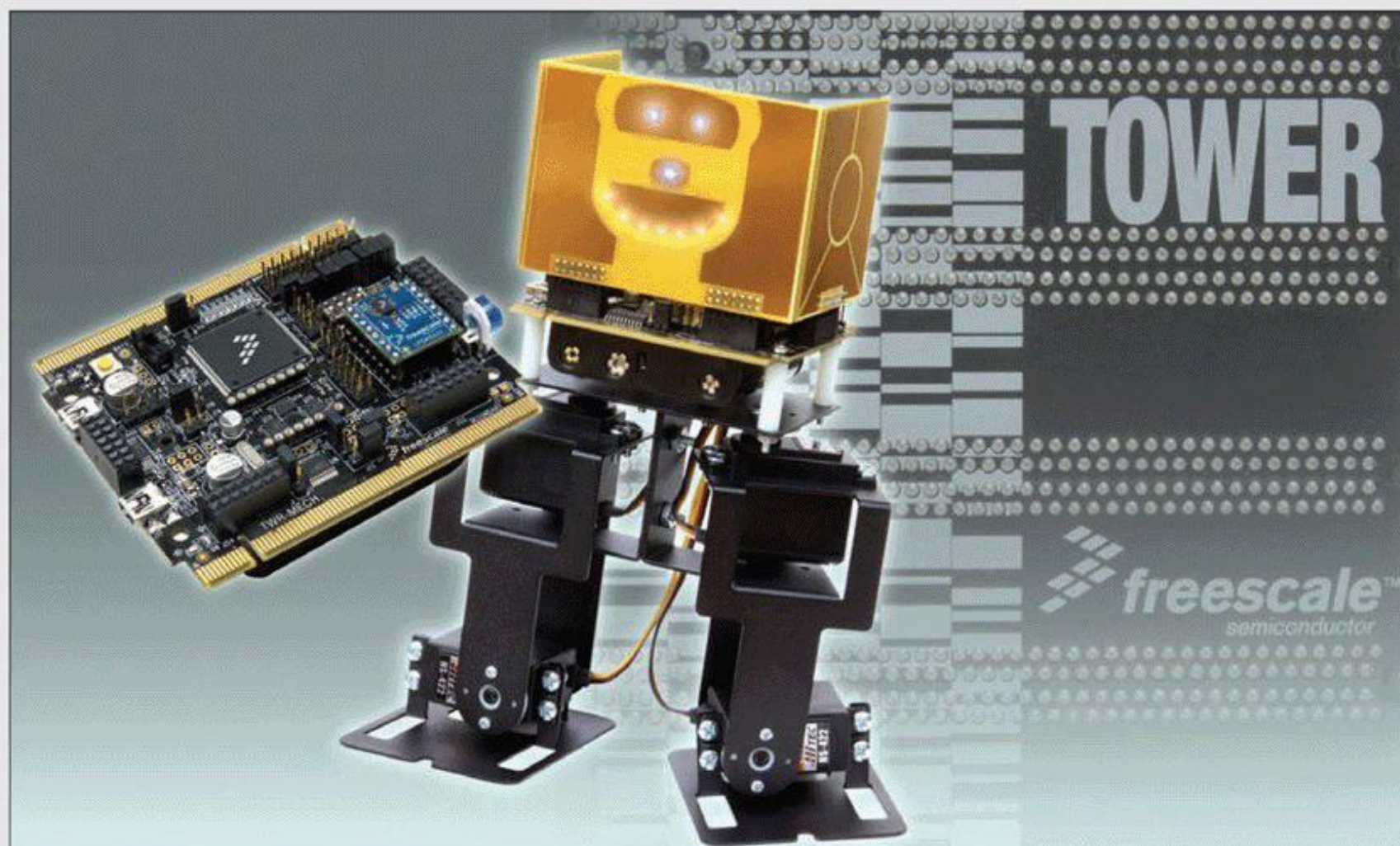
Eric's latest project is extending his RobotSee software platform (www.EMGRobotics.com) to work with the new robot from Freescale: the Freescale Tower Mechatronic Robot. RobotSee is a free, open source robotics toolkit that works with Windows, the Chumby, Android tablets, and phones, and Linux. As you might expect, RobotSee provides a vision toolkit that includes features such as face recognition. Moreover, the toolkit — which uses an easy-to-use language with similarities to both C and Basic — supports voice recognition, speech synthesis, GPS navigation, and even an interface to those affordable brain-machine interfaces that are on the market.

The Freescale Tower Mechatronic Robot (which I'll

refer to as simply 'Mech') is Freescale's first entry in the market targeting robotics enthusiasts. At \$199, Freescale is positioning the robot as the next step up from robots based on the Stamp or Arduino. The basic board — which goes for \$99 — uses a MC52259 32-bit microcontroller with 64K RAM and 512K Flash. It has space for two plug-in daughter boards, including a \$25 magnetometer or compass and a \$99 three-axis accelerometer. There's also a pair of USB connectors, analog and digital I/O, I²C, SPI, and even a legacy RS-232 port. Development software includes free versions of CodeWarrior and RobotSee. In terms of difficulty, RobotSee is just a bit more challenging than, say, programming the BASIC Stamp. CodeWarrior, on the other hand, requires modest familiarity with C/C++ programming, and doesn't come with built-in libraries for vision, speech, voice, and the rest.

Where Mech becomes interesting is when you drop in the 3" x 3" 1 GHz/1 GB Linux board on top of the basic board. The \$149 board has enough power to support, say, your own robot swarm. You can add a webcam for vision recognition, a WiFi or XBee transceiver for wireless communications, or your own custom sensors. As you can see, you'll want to perform a brain transplant as soon as you wrap your head around the Linux board.

Perhaps a crawler or R/C truck body would be more appropriate for the horsepower the Linux board brings to the table. Several years ago, I was forced to upgrade my hardware from lowly microcontrollers to a real time Linux board, at a cost of over \$1,000 for the bare board. At \$149, the Linux board is a bargain if you want to get into real time path finding or data fusion. If you already have a robot platform and know what you want to



FOR THE
ROBOT
INNOVATOR

SERVO
MAGAZINE

Published Monthly By
T & L Publications, Inc.
430 Princeland Ct., Corona, CA 92879-1300
(951) 371-8497
FAX **(951) 371-3052**
Webstore Only **1-800-783-4624**
www.servomagazine.com

Subscriptions
Toll Free **1-877-525-2539**
Outside US **1-818-487-4545**
P.O. Box 15277, N. Hollywood, CA 91615

PUBLISHER
Larry Lemieux
publisher@servomagazine.com

**ASSOCIATE PUBLISHER/
VP OF SALES/MARKETING**
Robin Lemieux
display@servomagazine.com

EDITOR
Bryan Bergeron
techedit-servo@yahoo.com

CONTRIBUTING EDITORS
Jeff Eckert
Tom Carroll
Dennis Clark
Fred Eady
Bryce Woolley
Gregory Intermaggio
David Ward
John Blankenship
Mike Jeffries
Jenn Eckert
David Geer
R. Steven Rainwater
Kevin Berry
Evan Woolley
Gordon McComb
Mark Mitchell
Samuel Mishal
Pete Smith

CIRCULATION DIRECTOR
Tracy Kerley
subscribe@servomagazine.com

**MARKETING COORDINATOR
WEBSTORE**
Brian Kirkpatrick
sales@servomagazine.com

WEB CONTENT
Michael Kaudze
website@servomagazine.com

ADMINISTRATIVE ASSISTANT
Debbie Stauffacher

PRODUCTION/GRAPHICS
Shannon Christensen

Copyright 2011 by
T & L Publications, Inc.
All Rights Reserved

All advertising is subject to publisher's approval. We are not responsible for mistakes, misprints, or typographical errors. *SERVO Magazine* assumes no responsibility for the availability or condition of advertised items or for the honesty of the advertiser. The publisher makes no claims for the legality of any item advertised in *SERVO*. This is the sole responsibility of the advertiser. Advertisers and their agencies agree to indemnify and protect the publisher from any and all claims, action, or expense arising from advertising placed in *SERVO*. Please send all editorial correspondence, UPS, overnight mail, and artwork to: **430 Princeland Court, Corona, CA 92879.**

Printed in the USA on SFI & FSC stock.



do with more capable hardware, I'd simply go for the two boards. On the other hand, if you're looking for an experimentation platform, then the Mech is worth considering. It has a dozen touch pads and does cradle the board(s) nicely. The LED eyes, nose, and mouth don't do anything for me, but again, it's handy to have LEDs already wired in circuit for testing purposes.

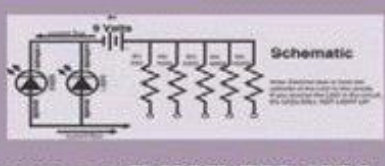
Whatever your robot platform, check out the RobotSee site. And, if you need more computational capacity, consider the fully configured Mech. **SV**



FUNDAMENTALS For The Beginner

Need the Basics?

Follow along with our series of articles which includes easy to understand graphics. Starting in the May 2010 issue.



Subscribe Today! (with optional digital back issue viewing.) Visit www.nutsvolts.com or call (800) 783-4624




BANEOTS.COM
970-461-8880

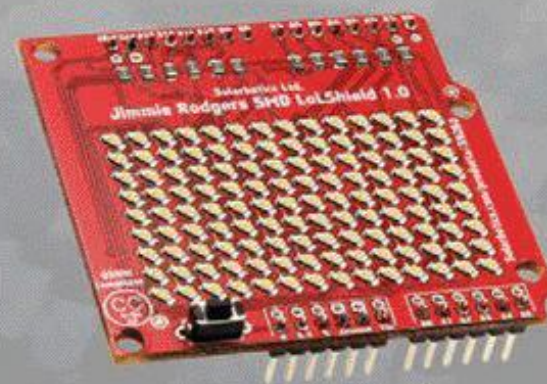
**MOTORS
GEARBOXES
WHEELS
AND MORE**



And on the eleventh day, Jimmie Rodgers said,
"Let there be light. Wait, scratch that. Let there be LOTS of lights.
Tiny ones. And while we're at it, cram 'em all onto an Arduino shield.
Yeah, that'd be pretty cool. We could use it as a display for stuff."



SKU: 39260 Price: \$31.50



The LoL Shield (Lots of LEDs) Shield gives you total control over 126 SMD LEDs to create a programmable display on your Arduino. Spell words, play games - you'll have a LoLin' good time.

SOLARBOTICS
www.solarbotics.com 1-866-276-2687

PS- And lo, the goofy messages displayed caused the Lord to smack his forehead in disgust.
PPS- Jimmie Rodgers may or may not have said those words. We're pretty sure he did. Probably.





Robytes

by Jeff and Jenn Eckert

Bird Flight Deciphered

In 2009, we introduced you to some flying penguins from Festo AG (www.festo.com), and last February, they came up with a handling assistant inspired by an elephant's trunk. This time, the company's fascination with animatronics has manifested itself in the form of SmartBird, an amazingly lifelike version of the herring gull. According to the company, "With SmartBird, Festo has succeeded in deciphering the flight of birds — one of the oldest dreams of humankind." The bird can start, fly, and land autonomously with no drive mechanism other than its wings. Maneuverability is enhanced by wings that twist at specific angles, as well as flapping up and down — a trick made possible by the



A real seagull eyes Festo's SmartBird with suspicion.

"active articulated torsional drive unit." The tail generates lift as well, in addition to its standard functions of elevator and rudder. In terms of the nuts and bolts of the thing, it is driven by a lithium polymer battery (two cells, 7.4V, 450 mA), a Graupner Compact 135 brushless motor, and a Texas Instruments MCU LM3S811 microcontroller. It's a bit larger than

the real thing, with a wingspan of 2 m (78.7 in) and a torso length of 1.07 m (42 in), vs. an approx. 56 in wingspan and 25 in body. It weighs significantly less, however: 0.45 kg/15.9 oz vs. up to 1.5 kg/3.3 lb. For more details, just point your browser to www.jkeckert.com/freedownloads/SmartBird.pdf and download the eight page brochure.

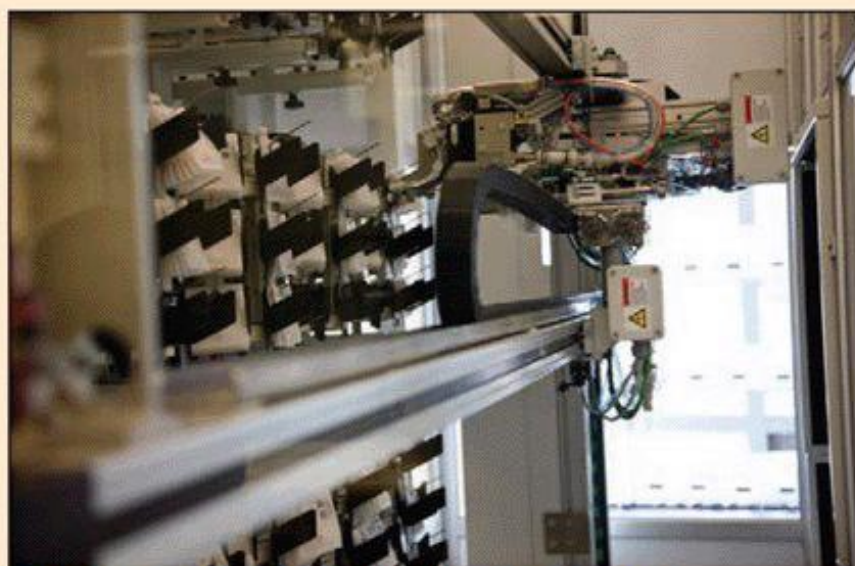
Bot System To Test Chemicals

It isn't news that robots are taking over more and more tasks that are dangerous or downright repugnant to humans, but the National Institutes of Health's Chemical Genomics Center (www.genome.gov) has given their machines a highly ambitious assignment. Its Tox21 program — undertaken in collaboration with several other agencies — will use a robot system to test 10,000 chemicals for toxicity. The results should provide information about which of these substances have the potential to lead to adverse health effects. The chemical under analysis will include a variety of compounds found in industrial and consumer products, food additives, and pharmaceuticals. And this could be only the beginning, as the list of suspect substances was referred to as the "initial" selection.

Tox21 has already tested more than 2,500 chemicals using various technologies, but the new system will speed up the process. As NCGC Director Christopher Austin noted, "The Tox21 collaboration will transform our understanding of toxicology with the ability to test in a day what would take one year for a person to do by hand."



A robot arm retrieves assay plates from incubators. (Credit: Maggie Bartlett, NJGRI.)



Robotic pill picker and plastic bags for medications. (Credit: Susan Merrell/UCSF.)

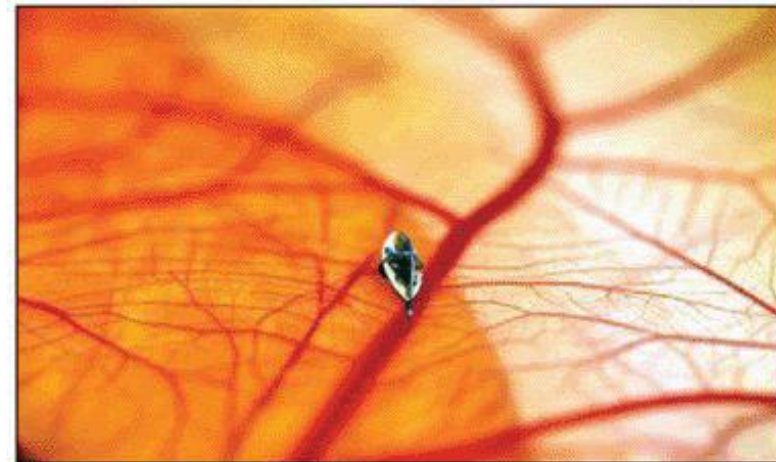
Drug-Dealing Automaton

Assuming the Tox21 program confirms that your favorite medication is acceptable, robotics may play an additional role in making sure you get the right prescription. A new pharmacy at the University of California, San Francisco (UCSF) Medical Center (www.ucsfhealth.org) is using robotic technology and electronics to prepare and track medications, thereby improving patient safety. Customers may not be aware of the bot's activities, as they are housed in a secure, sterile environment, but they are busy preparing oral and injectable medications, including toxic chemotherapy drugs. According to Mary Anne Koda-Kimble, dean of the UCSF School of Pharmacy, "Automated medication dispensing frees pharmacists from the mechanical aspects of

the practice. This technology, with others, will allow pharmacists to use their pharmaceutical care expertise to assure that patients are treated with medicines tailored to their individual needs." Reportedly, not a single error has occurred in the 350,000 doses prepared during the system's phase-in.

Something In Your Eye?

Lest we close out this month's offerings without providing something to raise goose bumps on the squeamish, consider a microbot developed at the Swiss-based Institute of Robotics and Intelligent Systems (IRIS, www.iris.ethz.ch). It's the latest in an effort to create tiny machines that can be injected into your eyeball and used to treat such conditions as macular degeneration. The electromagnetically controlled devices can move to a target location and remain there for months, releasing drugs. The bots can also install a biodegradable drug capsule and then be removed with a magnetic needle. If you haven't squirmed enough, visit <http://bcove.me/bb27w039> for a video of the bot creeping around inside the eye of a dead pig.



Microbot sitting on a vein in a chicken embryo, used as a model for retinal veins. (Credit: IRIS.)

Reconfigurable Bot Goes Commercial

With funding from the National Science Foundation, a reconfigurable modular robot invented at the University of California, Davis (www.ucdavis.edu) is headed for commercial development. The iMobot, developed by Graham Ryland and Harry Cheng, is particularly useful as a teaching tool, and they say there are currently no commercially available research-grade robots. It also has potential applications for prototyping complex assemblies, and they speculate that it could eventually be configured for search-and-rescue operations.

Each module has four degrees of freedom, with two joints in the center and a wheel on each end. It can roll on its wheels, crawl like an inchworm, or raise one end of its body and pan around (as, e.g., a camera platform). Individual modules can be combined into larger assemblies such as snakelike robots or larger, wheeled machines that can roll over smooth terrain.

By using an off-the-shelf bot like iMobot, it is believed that researchers can work in such areas as AI, robot collaboration, and reconfigurable and adaptive systems without having to develop proprietary hardware systems.

The new company, Barobo Inc. (www.barobo.com), received an initial grant of \$150,000 for a six month period, with an additional \$500,000 potentially available. The bot should be on the market by the end of the year. For a vid, go to YouTube and search iMobot. **SV**

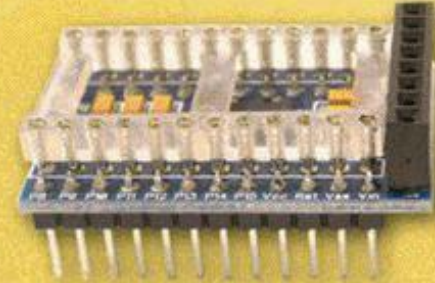


iMobot rolls, crawls, and creeps.

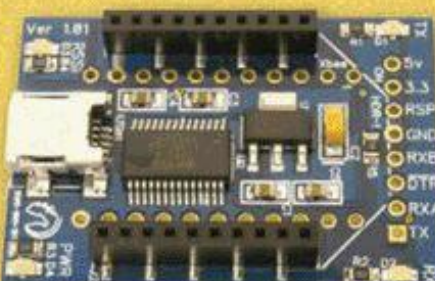


GREAT FOR
Wireless Data Links,
Mobile Robots,
Alarm Systems, Sprinkler
Systems & Weather Stations. Anything Remote
that needs a wireless Link. If you have a BASIC
Stamp, now you can program it wirelessly!

FlashFly Serial Wireless Kit **\$44.95**
FlashFly USB Wireless Kit **\$47.95**
Program BASIC Stamps Wirelessly,
or use for a wireless data link.
See our site for more info.



USB Stamp Adapter Module **\$15.95**
Use with FlashFly and program your USB Boe-Bot
wirelessly or it's Great for breadboarding.
(LED PWR indicator and 5.0V regulator built in)



XBee USB Explorer Module **\$19.95**
Experiment with XBee modules via USB.
(3.3V regulator built in)



10 Pin 2.0mm XBee **\$0.95**
Female HDR Sockets

Visit our online store at: www.bluewolfinc.com for more information and other items
Use Coupon Code: S515 for 5% Discount on orders over \$15.00



GEER HEAD

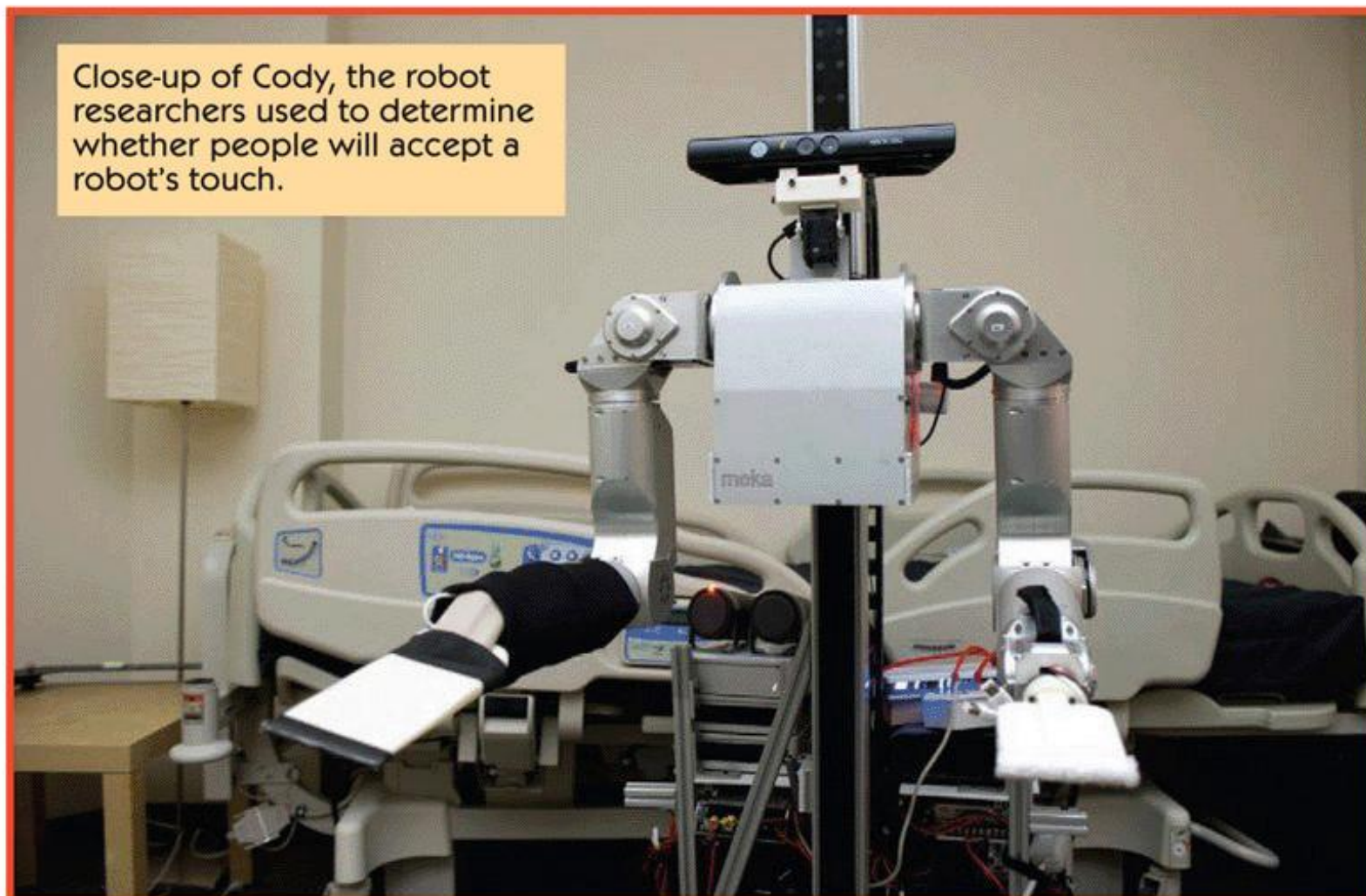
by David Geer

Contact the author at geercom@windstream.net

A Robot's Touch

He may not be the lead singer from Def Leppard, but he still "wants to touch you." Cody — a nurse assistant caregiver robot — wipes down subject's forearms to clean them, or touches them to offer comfort in a critical experiment with a surprising conclusion.

Close-up of Cody, the robot researchers used to determine whether people will accept a robot's touch.



The research is motivated by a pervasive nursing shortage globally which — according to the researchers Tiffany Chen, Chih-Hung King, Charles Kemp, and Andrea Thomaz — is forecasted to increase in coming years. It should be noted that due to advances in health care that are increasing longevity, plus the retirement of the large baby boomer population, there will be a larger patient population requiring nursing care.

Because bathing patients takes up a significant amount of a nurse's work day, teaching a robot to perform the task would free up a great deal of time for more critical services to patients. This experiment gave the roboticists insight on the type of touch people appreciate more when it comes from a robot. If the robot is touching them to provide a service (clean the forearm), the touch is welcomed. If the touch is to transfer comfort, the subjects do not appreciate it as much. This parallels research conducted on human nurse and patient touch. The researchers also examined whether a warning as to the purpose of the touch beforehand was helpful.

The Research and Reasons

Georgia Tech researchers Tiffany L. Chen, Chih-Hung King, Charles C. Kemp, and Andrea L. Thomaz produced research based on the touch of the Cody robot on 56 research subjects at the world famous research university.

The purpose of the experimentation was to develop a robotic nurse's assistant that could perform duties such as giving patients baths while in their beds. Goals of the research included testing and comparing nurse and patient interaction with robot and patient interaction, for efficacy and application.

From the nurse and patient touch studies, two particular types of touch resurfaced: instrumental touch to perform a required medical task or treatment and affective touch to comfort patients. With human nurses, patients are more receptive to touch for instrumental purposes than for the purpose of comfort, according to the several studies.

With the robotic nurse assistant, the researchers sought to determine whether the patients had the same reaction

to the two different purposes of touch. The scientists also sought to learn whether patients preferred to be warned verbally by the robot that they were about to be touched and what the purpose of the touch was.

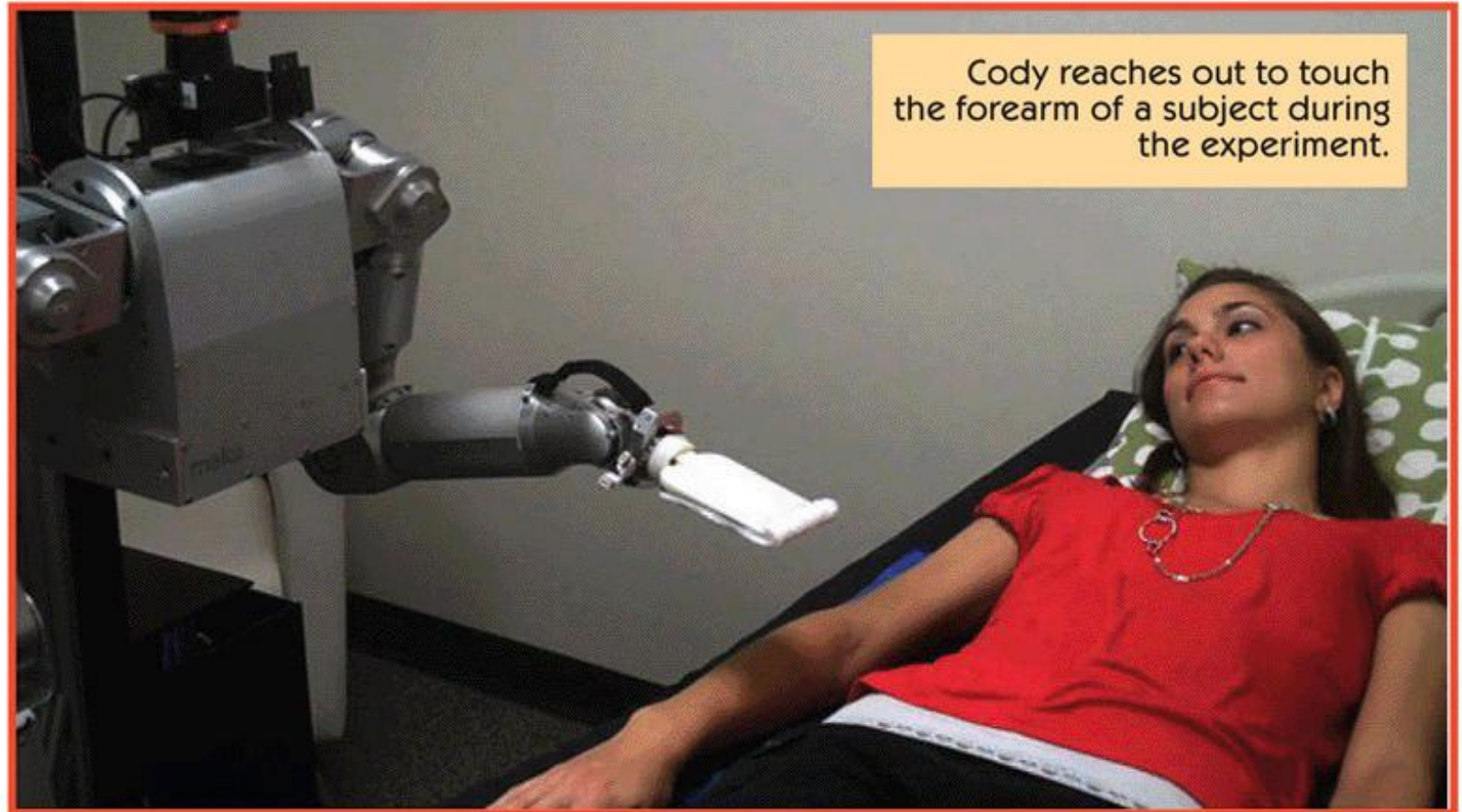
The Robot

Cody is a mobile two-armed robot created for testing the robotic nursing assistant paradigm. Cody's mobility is enabled by its Segway base. Cody's manipulators consist of MEKA Robotics arms with each one actuated by seven series elastic actuators at each of its seven joints for seven degrees of freedom and smooth actuation. Each joint also has virtual visco-elastic springs to ensure that the robot's hands stop where they should.

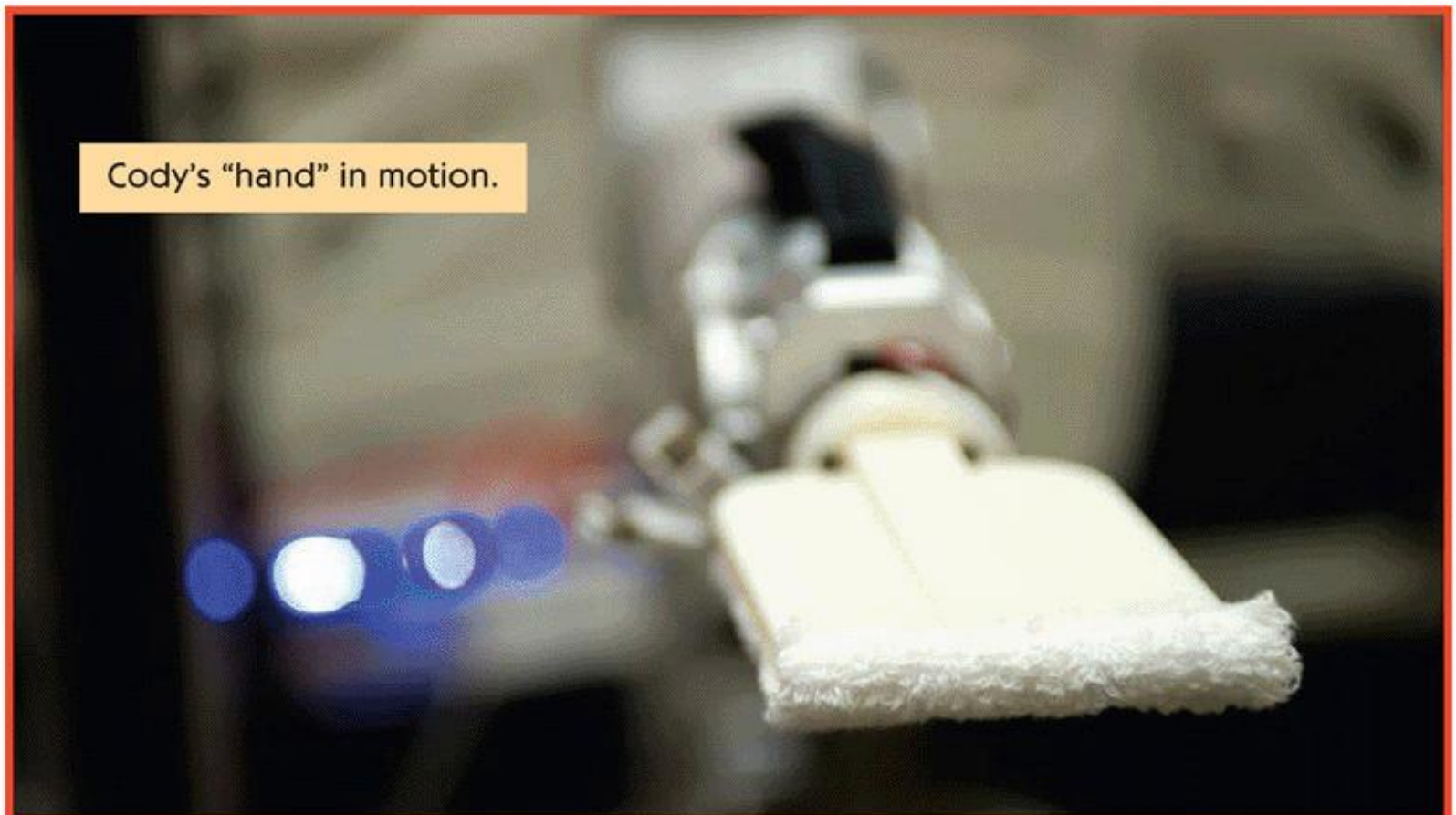
The wrists are equally equipped with multiple degrees of freedom via six-axis force sensors. The hands — which look like spatulas — were created using a 3D printer. The robot touches subjects with cuttings of towels that have been wrapped around the hands and secured.

The body of the robot consists of a torso with an RX-28 Robotis servo on top of that. The robot's intelligence consists of two Linux computers running the popular free Ubuntu Linux distribution and software the roboticists coded in with the Python programming language. In order for the robot to touch patients precisely, the researchers installed a tilting Hokuyo UTM-30LX and a Point Grey Firefly camera to gather laser range data. These are contained in an XBOX 360 housing.

The robot system includes an interface on a remote computer so



Cody reaches out to touch the forearm of a subject during the experiment.



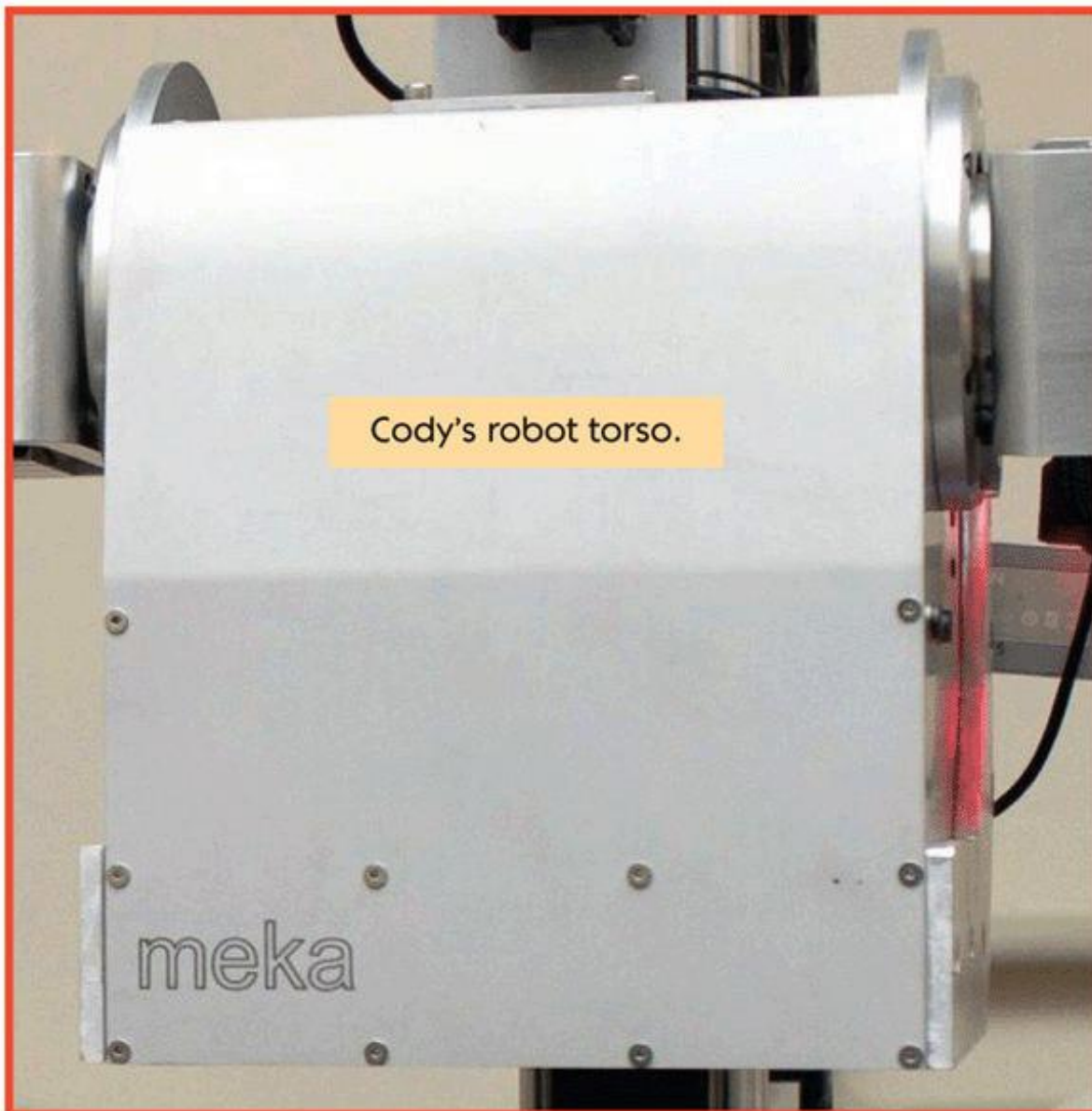
Cody's "hand" in motion.



Cody's robot housings and electronics (including power to the Segway), its computers, and controller box.



Cody's XBOX 360 robot head containing the cameras and laser sensor.



Cody's robot torso.

the operator can choose the area of the subject that they wish the robot to wipe down. Then, the robot performs the wiping in a fully autonomous fashion.

A number of safety features are provided to keep human subjects from harm when the robot is touching them. The robot is mounted with a run stop button or kill

switch to stop the robot if necessary. The low stiffness in the robot's arms means that impact with the human subject is lessened. The robot controller also limits the downward force of the hand when wiping. Operators can manually command the robot to stop if the force measured in the robot's wrist sensor exceeds a certain value.

A Robot's Touch

The robot starts the touching behavior by making initial contact with the forearm, then moving the end effector and towel along the forearm. In the experiment, the researchers tested two hypotheses. First, the patient will accept the robot-initiated touch more if it is seen as instrumental to their care, rather than have it be affective (comforting). Second, subjects will find the touch more favorable when the robot gives them a verbal warning first.

For the experiment, the researchers created a mock hospital room complete with a real hospital bed. Fifty-six subjects from Georgia Tech between the ages of 18 and 29 (half male and half female) participated in the testing. Several tools were used to gauge the subject's reactions to the robot. A pre-task and post-task questionnaire was given. Subject's emotional states were measured using the Self-Assessment Manikin (SAM) and Positive and Negative Affect Schedule (PANAS) tests.

The researchers also queried the participants using a customized Likert scale questionnaire. For this, the question set included things relative to the researcher's specific hypotheses. These included questions as to whether the subject was confused as to why the robot was touching their arm; whether they enjoyed the robot touching them or were scared by it; whether they were reassured by the robot touching them; whether they felt it was necessary for the robot to touch their arm; whether they would let the robot touch them again; and whether

Resources

Cody robot video
www.digitallounge.gatech.edu/healthandeducation/index.html?nid=64852

Georgia Tech Healthcare Robotics Lab
<http://healthcare-robotics.com>

The Segway, used for the robot's base and mode of transportation
www.segway.com

MEKA Robotics, source of Cody's arms
<http://mekabot.com>

RX-28 Robotis servo
www.trossenrobotics.com/dynamixel-rx-28-robot-actuator.aspx

The Ubuntu Linux distribution used in Cody's computers
www.ubuntu.com

The Python programming language official site
www.python.org

Cody's Hokuyo range finder
www.hokuyo-aut.jp/02sensor/07scanner/utm_30lx.html

Cody's Altec Lansing speakers that he uses to communicate.

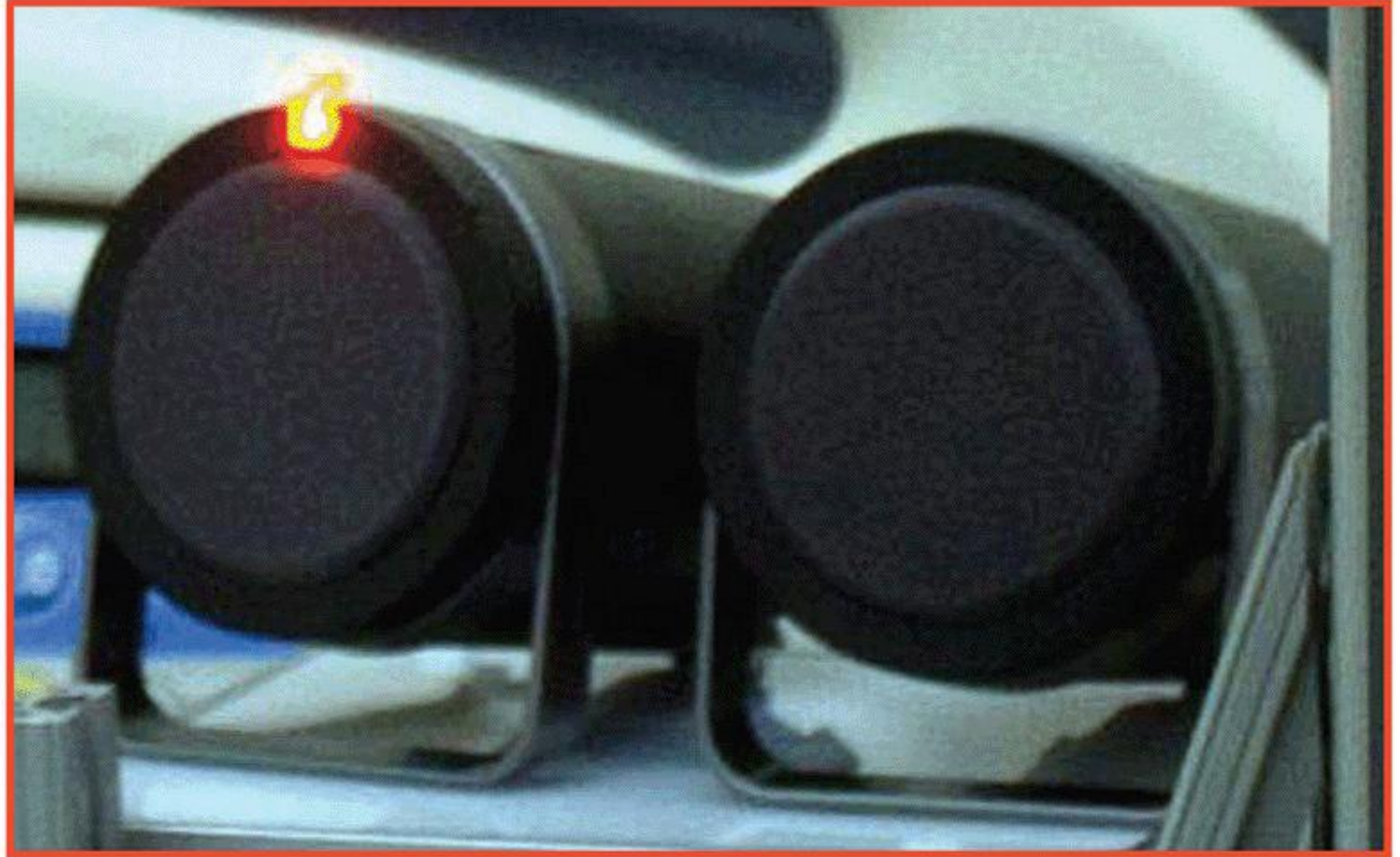
they would have preferred the robot not to touch them. Participant's faces were monitored by a camera, as well.

While the subjects did like to be touched for instrumental reasons (which agreed with the researchers' first hypothesis), the subjects did not like to be warned before being touched (which was against the researchers' second hypothesis). This will likely play a key role in future research and development.

Conclusion

The Georgia Tech researchers are confident that the Cody research could lead to a real robot assistant to make nurses' lives easier and to help take up the slack created by the nursing shortage.

The research is amply supported by Hstar Technologies, an NSF Graduate Research Fellowship Program, Willow Garage (commercial robotics firm), and NSF Grant IIS-0705130. **SV**



www.POLARISUSA.com

1000' Long Range
100% Digital Video Links
H.264 or MPEG4 Compression Available!



MB-830B-OS-12

1/3" Color Infrared Board Camera
420TVL, 12mm Lens, 12VDC 110mA



WRL-5800VK

5.8GHz Digital Audio/Video
Transmitter/Receiver Kit, 755 Ft. Range

Need a Mega Pixel Camera?
Check out our Website!



Lens not included

BC-610B

1/3" B/W Brick Camera
600TVL, 0.001Lux
12VDC 150mA



MBC-550DC

1/3" Color Board Camera, 550TVL, Day/Night
4.0-9.0mm Varifocal Auto Iris Lens, 12VDC

Actual images may vary!

Give Your Robots Eyes!

PolarisUSA Video

3158 Process Drive Norcross, GA 30071-1602
Local 678-405-6080 Toll Free 800-308-6456

Our resident expert on all things robotic is merely an email away.
roboto@servomagazine.com

Tap into the sum of *all human knowledge* and get your questions answered here! From software algorithms to material selection, Mr. Roboto strives to meet you where you are — and what more would you expect from a complex service droid?

ASK MR. ROBOTO

by
Dennis Clark

Update from Last Month's Column on Optical Mouse Navigation

When last we met, I was going on about using a PS2 mouse for assistance with robot navigation. I was pretty sure that you could do some simple math with rectangular to polar coordinate transforms to get distance and heading information that would be more accurate than using simple dead-reckoning. I'm still pretty sure this is possible, but ... Mr. Roboto has to-date failed to get a PS2 mouse to listen and talk back. So, either my mouse is not fully functional, or, the lack of a datasheet for this model has put a serious crimp in my understanding of the protocol. Regardless, I have been humbled by a lowly mouse!

I don't give up easily, but I don't bang my head against the wall either. Since using PS2 mice is a bit tricky, they are getting old and hard to find, cost \$5 or so, and finally, since there are Avago sensors available that you can talk to using simple SPI hardware, I'm going that route. Here is my plan and **Parts List** to get an optical mouse sensor to be a navigation aid:

PARTS LIST

Part #	Digi-Key	Cost
Avago ADNS-5050 Optical Sensor	516-2261	\$1.60
Avago ADNS-5100-001 Trim Lens	516-2300	\$0.57
Avago HLMP-ED31-SV00 Red LED	516-1372	\$0.77
Avago HLMP-5029 LED Clip	516-1395	\$3.78 (for 10)

As you can see (except for having to buy 10 of the LED clips), I could get the entire sensor setup for less than a cheap mouse — much less. Plus, I get the ability to take low resolution pictures and do lots of cool stuff with the chip directly. I'm going to mount this on a small proto board and get back with you next month with a status report on

how well this works. I'm also going to recycle this optical mouse (with obsolete parts in it) before I lose any more sleep!

Before I go on, I'd like to thank the readers for some truly interesting questions and problems to solve. While I love to solve problems and help with your robotic construction questions, I am still only human. Every so often, I receive a question that — if I were to completely answer it — I would have to go back to grad school and write a dissertation to cover it. I do enjoy a challenge, but please, keep your questions focused on a single issue or problem if possible. If it takes you three pages to cover all the questions, it could take me months to answer them!

Q . I want to use a laptop with one or more USB joysticks to control my underwater ROV. But, I have no idea how to code the actual reading of the joystick axes and buttons. Once I can get the raw data into a C program (preferably using gcc and public libraries), I can write the code to convert axis movement in forward, reverse, rotate, up, down, arm move, ballast, etc. Can you point me to the interface library and a sample interface code?

— Mike

A . A joystick is a USB HID device and every computer has libraries to handle communications with a HID device. The tome on handling this interface is, of course, Jan Axelson's *USB Complete*. Get a copy of this book or check it out from your local library; everything you need is in there. If you are an old hand at PC programming, then you will have no problem interfacing with a set of USB joysticks.

There are a few sites out there that can help without the book of course, but you'll get a lot more mileage from your work with *USB Complete*. Start by looking here:
<http://www.lvr.com/hidpage.htm>.

Q. I want to build my first robot, but I don't know where to start. I've looked at motors and chips to power them. I've looked at sensors — there are a LOT of sensors to choose from! There's one for just about anything I would want to do. My biggest question is, what controller should I use? There are Netbooks, PICs, then Atmel, ARM, Motorola, and more. Where do I start?

— Tom

A. Ask me any question but that one. Picking the "best" microcontroller is practically a religious question in the robotics community. Everyone has their own opinion and very good reasons to support their choice. Over the last 10 years, I've changed my mind many times. I started out with Parallax Stamps back in <mumble> when I was getting started again in robotics. They were cool after coming from using Z-80s on wire-wrapped motherboards using assembly language. I wanted more RAM and more controls, so I moved to Motorola 68HC11s and their ilk. Those were nice for a while, but I outgrew them, as well. I then went to the PIC16Cnnn and then the PIC18Fnnnn parts using CCS C and Microchip C compilers. Others really like the PICBASIC PRO compilers, which are also very good.

Then, I moved to the Atmel AVR series. At first, I used BASCOM AVR as a teaching platform for classes because they did a good job abstracting the hardware layer. This is useful for teaching those new to the art of programming. Next, I discovered avr-gcc and avr-dude which ran on Macintosh and Linux, and freed me from Windows platforms. Shortly thereafter, the Arduino environment — also OS agnostic — came out for the Atmel ATMEGA chips, which is fantastic as a learning/teaching platform. I have not yet done much with the ARM platform. So far, I find the tools and IDEs difficult to use and even more difficult to explain to others. That is starting to change, and I'm keeping an eye on them.

Recently, I've been attracted to the PIC24FJ and PIC24K series of 16-bit chips. They are fast and easy to use. They still need somewhat expensive programmers (\$190 or so for an ICD3), but the compiler is free. What is even better is that Microchip is coming out with MPLAB-X which is OS agnostic. So again, I'm interested because it means I don't have to develop on a Windows platform. As I outgrow a particular platform, I migrate to a higher powered, faster one.

The processor to choose for your hobby will depend upon your experience with programming and your needs for speed. The development platforms that take great lengths to abstract the hardware (CCS C, PICBASIC PRO, BASCOM/AVR, Arduino) will be very attractive to programmers who don't understand hardware and those new to the

craft. You will outgrow these only if you want to get more involved directly with hardware. If you don't want to invest a lot, then you will want to use open source compilers and IDEs (Integrated Development Environments) like Arduino and avr-gcc since you can get cheap hardware, and sometimes you don't need any programmers to program the parts. If you are going to be doing vision and object recognition, then you should start with the Netbook or laptop approach. These are the only platforms that will have the computing power you'll need.

For new roboticists or embedded enthusiasts, I recommend the Arduino. Cheap hardware, free development environment, and LOTS of user group support and example programs. You can then migrate to open source gcc compilers to get more speed, power, and control. Gcc is also often the environment of choice when you move to full-power laptops and the like to get really powerful robots. In the end, it is a personal decision based on your comfort level with working close to the hardware and how you choose to do it.

Q. I want to put a graphical display on my robot controller for a fancy interface remote control. Is there such a thing? And more importantly, can I afford it? Do you have any suggestions?

— Tony

A. You bet there are, and you can afford them too! Just for fun, I got a couple of different units and built the kits. The first one I found was at SparkFun. No surprise there. They always seem to have cool stuff. The kit that I got was the SparkFun Color LCD Shield LCD-09363

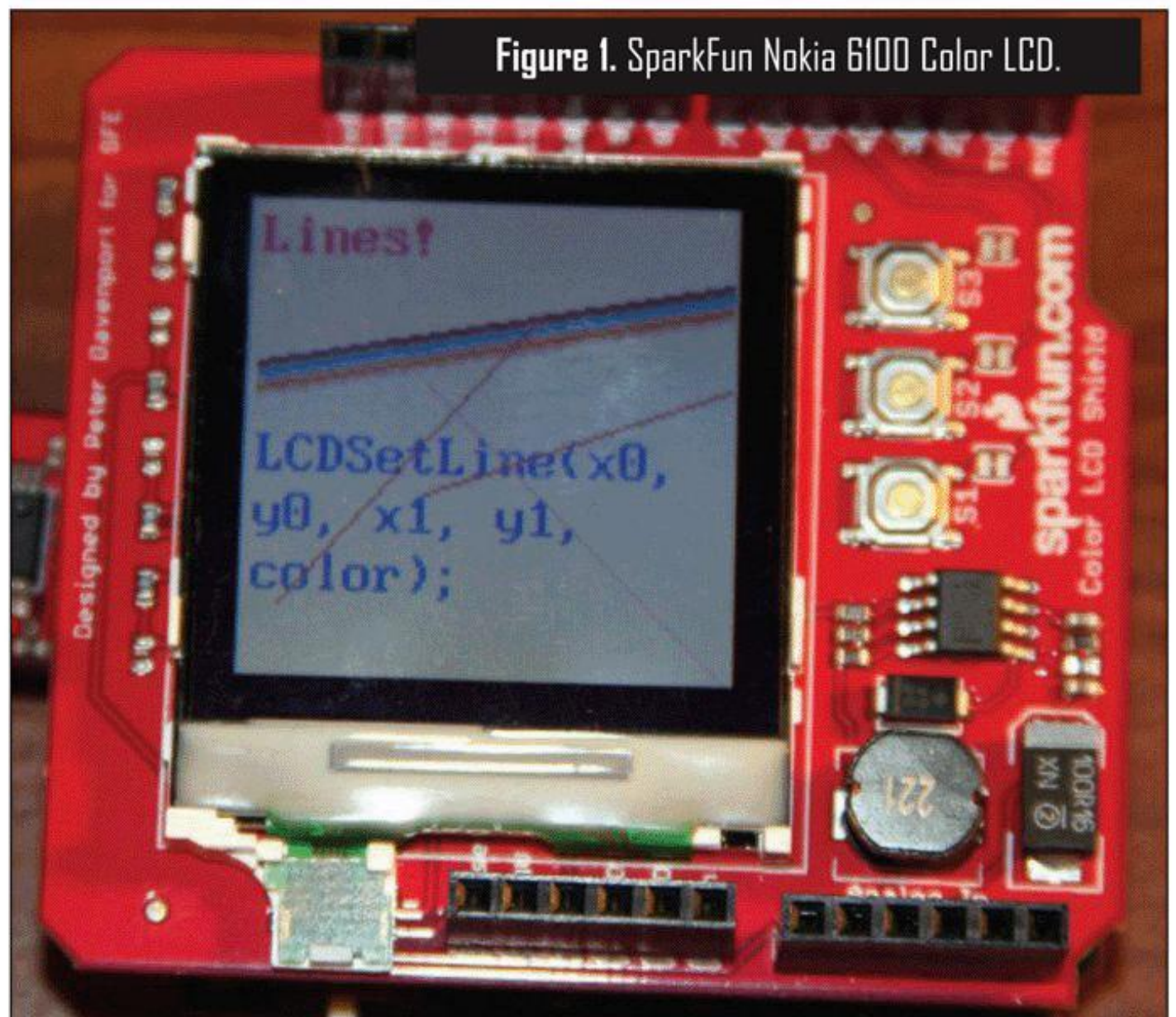


Figure 1. SparkFun Nokia 6100 Color LCD.

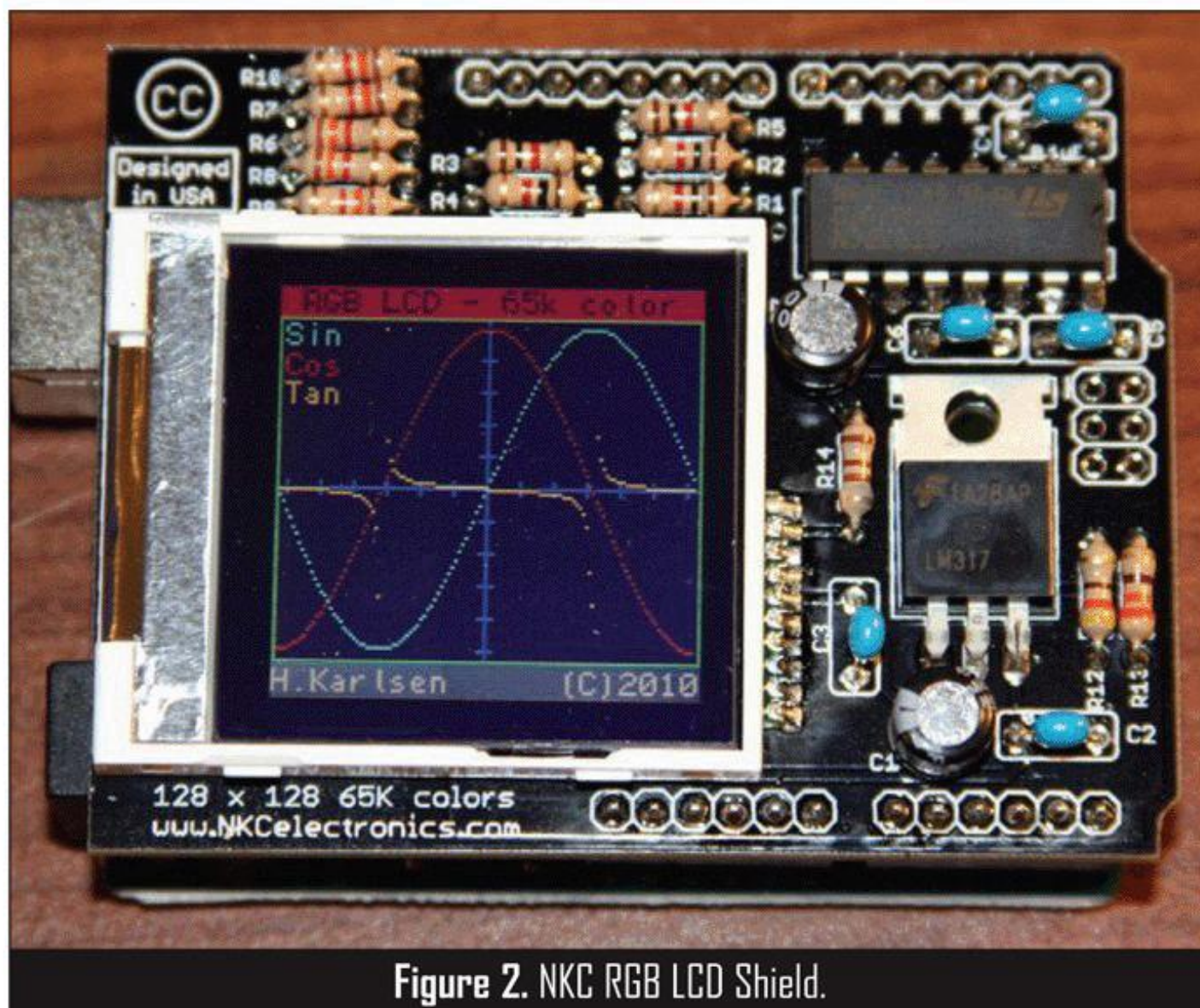


Figure 2. NKC RGB LCD Shield.

(Figure 1). SparkFun sells this for \$40 and they have links to libraries for it. One library is avr-gcc friendly C. The other two are — as you can imagine — Arduino-based libraries

(Arduino calls expansion boards *shields*). The Nokia displays 4096 colors and is a 128 x 128 pixel screen.

Of the two color LCD boards I played with, this one has a slightly larger display (1.2" square). The native avr-gcc library runs faster than the Arduino ones, but the Arduino libraries are quite usable regardless. SparkFun also sells the LCD displays (Nokia 6100) individually, so you can build your own boards using just the display and their example schematics if you want. The SparkFun board comes fully assembled with three pushbuttons and a working backlight. Look for it at www.sparkfun.com.

The other color LCD display I found was also an Arduino shield — the NKC Electronics RGB LCD Shield for Arduino 65K Color Kit ARD-0065. This LCD display looks similar, but is quite a bit different from the SparkFun unit. The display is smaller (1" x 1"), there are no surface-mount parts, and the unit comes as a kit that needs to be assembled.

The price is \$20 — which is cheaper — but you have to build it. So, pick your poison.

This LCD display also has an Arduino library written for it and it comes with a WAY more fancy demo program (see Figure 2). There are no I/O buttons on this board, and it too has a nice bright backlight. Maybe it was just the demo, but the NKC board seemed to have brighter color. I'll have to play with them to see if this is really true. Like SparkFun's Nokia 6100 LCD, this display is 128 x 128 pixels. Unlike the Nokia display the Phillips PCF8833 compatible display, is a full 16-bit color device. NKC also sells just the LCD display for your own designs.

I found both of these boards to be good looking and easy to read. The SparkFun board comes pre-built with buttons on it; the NKC board has to be built and has a slightly smaller screen, but more colors and it's cheaper. They both work great, however.

Well everyone, have fun and keep building robots. I'll be back next month with an update on using optical mouse sensors for navigation — I should be able to get that project working at least! Until then, as usual, if you have any questions for Mr. Roboto, feel free to email me at roboto@servomagazine.com and I'll be happy to try to answer them! **SV**

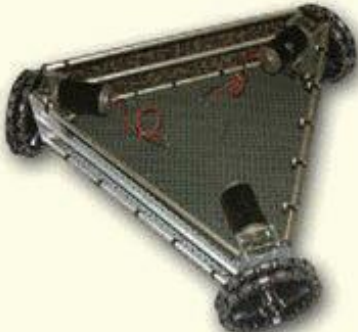


AndyMark
Inspiring Mobility

**Specializing in Unique Wheels,
Gearboxes, Aluminum Sprockets
and Drive Bases**



**6" Aluminum Dualie
Omni Wheel**



Tri- Lambda Drive Base
Omni-directional drive
system kit.



Aluminum Sprockets



8" Mecanum Wheel

AndyMark, Inc.
sales@andymark.com

Toll Free: 877-868-4770

www.andymark.com

Das Blinkenboard
Not just for blinken LEDs.

Can be used to operate
motors, solenoid valves,
relays, or any DC load up
to 24V/500 mA per channel!

<http://store.servomagazine.com>
or call 800 783-4624



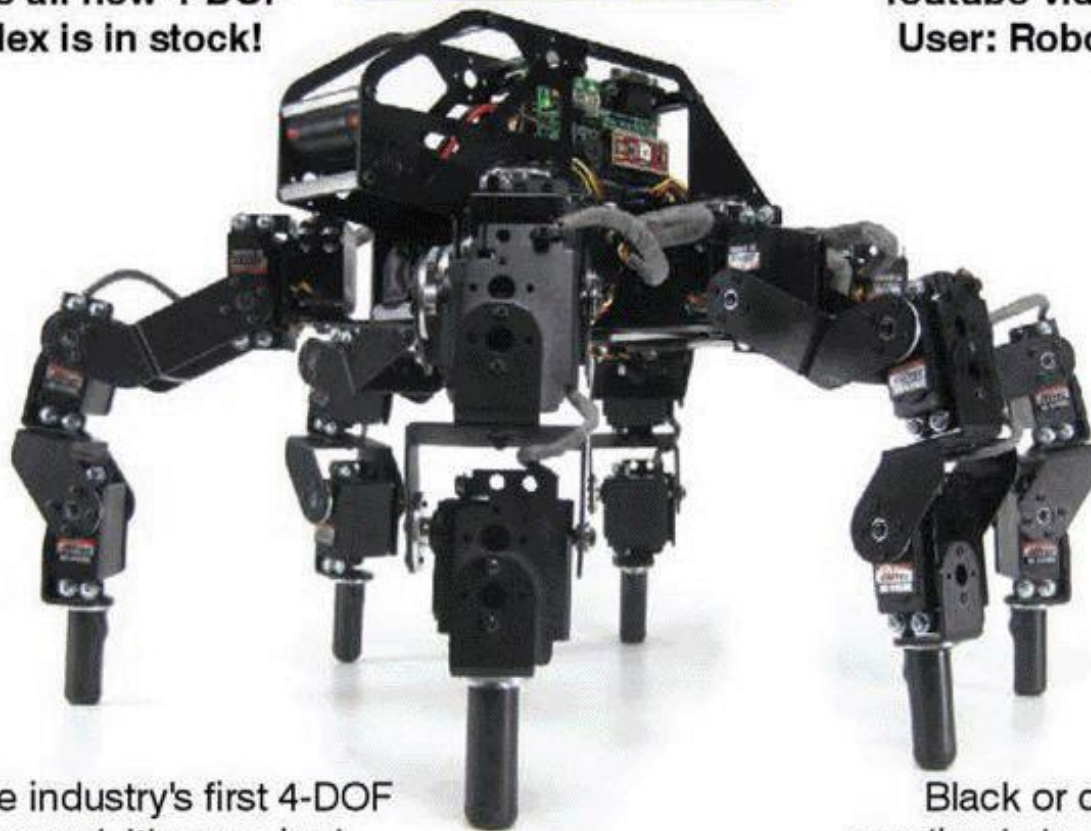


The Lynxmotion Servo Erector Set Imagine it... Build it... Control it!

Featured Robot

The all new 4-DOF
T-Hex is in stock!

Youtube videos
User: Robots7



The industry's first 4-DOF
hexapod. It's amazing!

Black or clear
anodized chassis!



Biped Nick



Biped Pete



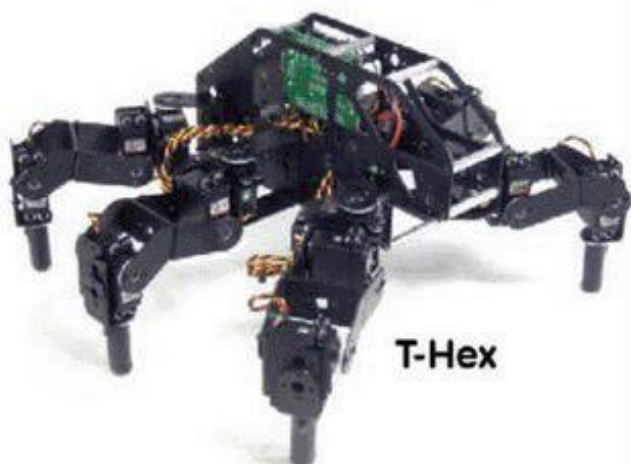
Biped Scout



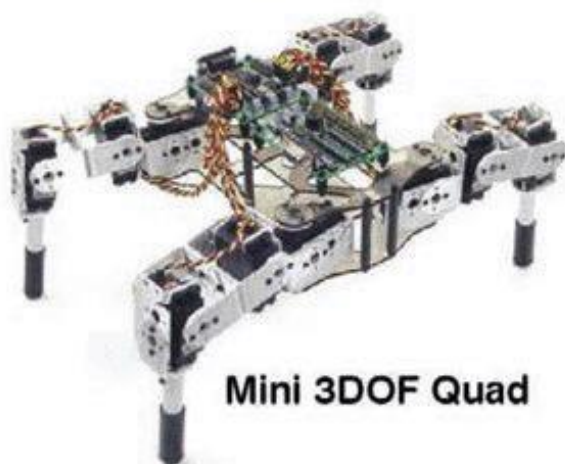
Biped 209



Walking Stick



T-Hex



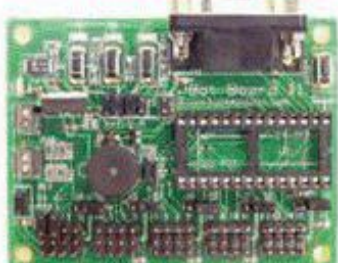
Mini 3DOF Quad

With our popular Servo Erector Set you can easily
build and control the robot of your dreams!

Our interchangeable aluminum brackets, hubs,
and tubing make the ultimate in precision
mechanical assemblies possible.



All New ARC-32 - \$99.95
32 Channel Servo/Microcontroller.
USB Programming port.
32 bit Hardware based math.
Program in BASIC, C, or ASM
Servo and Logic power inputs.
Sony PS2 game controller port.



Bot Board II - \$24.95
Carrier for Atom / Pro, BS2, etc.
Servo and Logic power inputs.
5vdc 250mA LDO Regulator.
Buffered Speaker.
Sony PS2 game controller port.
3 Push button / LED interface.



SSC-32 - \$39.95
32 Channel Servo Controller.
Speed, Timed, or Group moves.
Servo and Logic power inputs.
5vdc 250mA LDO Regulator.
TTL or RS-232 Serial Comms.
No better SSC value anywhere!

We also carry motors, wheels, hubs, batteries, chargers,
servos, sensors, RC radios, pillow blocks, hardware, etc!



Visit our huge website to see our complete line of
robots, electronics, and mechanical components.



CH3-R



Biped BRATs



Phoenix



Images represent a fraction of what can be made!

www.lynxmotion.com

The SES now has over 200 unique components!

EVENTS

Calendar

ROBOTS.NET

Send updates, new listings, corrections, complaints, and suggestions to: steve@ncc.com or FAX 972-404-0269

Know of any robot competitions I've missed? Is your local school or robot group planning a contest? Send an email to steve@ncc.com and tell me about it. Be sure to include the date and location of your contest. If you have a website with contest info, send along the URL as well, so we can tell everyone else about it.

For last-minute updates and changes, you can always find the most recent version of the Robot Competition FAQ at Robots.net: <http://robots.net/rcfaq.html>

— R. Steven Rainwater

JUNE

2-4 ION Autonomous Lawnmower Competition *Beavercreek, OH*

Basic and advanced autonomous lawn mowing robots try to be the fastest and most accurate at mowing a lawn while avoiding stationary and moving obstacles.

www.automow.com

3-6 AUVS International Ground Robotics Competition

Oakland University, Rochester, MI

Autonomous ground robots must navigate an outdoor obstacle course within a prescribed time limit, while staying within a speed limit.

www.igvc.org

10-12 National Underwater Robotics Challenge *Chandler, AZ*

Simulated mission to another star system. Robot probe must melt through the surface of an ice planet, descend into the ocean below, deploy a data bouy, and set up a nuclear powered AUV power station, while gathering samples of any life forms found. www.h2orobots.org

11 Pobot Junior Cup *Centre International de Valbonne Sophia-Antipolis, France*

Events include maze solving for LEGO NXT robots and a pick-n-place contest.

www.pobot.org

16-18 MATE ROV Competition *NASA JSC Neutral Bouyancy Lab, Houston, TX*

ROVs complete tasks related to a simulated deep-water oil spill including removal of damaged riser pipes, well capping, and collecting water or biological samples.

www.marinetech.org/rov_competition

22-27 Eurobot *Astrakhan, Russia*

This year, autonomous robots compete against each other to stack up "pawns" on a field roughly based on a chess game.

www.eurobot.org

25-26 International Autonomous Robot Contest *San Diego County Fairgrounds, San Diego, CA*

Autonomous robots navigate around fixed obstacles.

www.iaroc.org

JULY

4-10 RoboCup Robot Soccer World Cup *Istanbul, Turkey*

All the usual robot soccer events include software simulations, small robot soccer, legged robot soccer, even humanoid robot vs. human soccer.

www.robocup.org

8-12 BOTBALL National Tournament *Orange County, CA*

Student-built autonomous robots move black and white balls on a game board.

www.botball.org

11 RobotRacing *University of Waterloo, Ontario, Canada*

Autonomous robot car races.

www.robotracing.org

12-17 AUVS International Underwater Robotics Competition *SSC Pacific TRANSDEC, San Diego, CA*

Autonomous underwater robots must complete a course with various requirements. The contest is being renamed to the "International RoboSub Competition" this year. www.auvsifoundation.org/AUVSI/FOUNDATION/Competitions



Now Serving Europe with Optimized Logistics

- Currency in EURO
- 3 day shipping to 70% of the territory
- Service in French and English
- Competitive shipping rates
- Huge product selection

One Global Door for Manufacturers

VISIT: **www.RobotShop.com** Shipping Worldwide

Robotics at your service! TM

NEW PRODUCTS

TOOLS & KITS

Electronics Explorer Kit

Digilent, Inc., announces the launch of the Electronics Explorer Board (EE Board) — a powerful, all-in-one personal circuit design station. The EE Board is an affordable product designed for students, hobbyists, and engineering professionals, and includes everything needed to build and test a wide range of analog and digital circuits right on the desktop, without the need for any other equipment.

Priced for personal ownership, the EE Board provides students unlimited access to real hardware and software tools to design, simulate, test, and observe real circuits. Without the restriction of a traditional university lab setting, students can spend more time experimenting with real circuits and building knowledge through direct observation and experience.

The EE Board incorporates a solderless breadboard, a triple-output programmable power supply, a suite of test and measurement devices, and a high speed USB2 port to connect to the users' own PC. The station works in conjunction with WaveForms™ — a free software product that provides intuitive and easy to use interfaces to EE Board test and measurement devices.

Using only the EE Board and accompanying instrument pack, a wide variety of circuits can be constructed and tested by connecting to any one of the six included test and measurement devices, including: four-channel 40 MSa oscilloscope; two-channel arbitrary waveform generator; triple-output power supply (two programmable); four-channel voltmeter; two programmable reference voltages; 32-channel logic analyzer; 32-channel pattern generator; and an assortment of digital I/O devices. Users can complete designs anytime, anywhere.

WaveForms easy to use interfaces allow users to acquire, store, analyze, produce, and reuse analog and

digital signals. A high speed USB2 connection ensures all instruments respond in near real time. WaveForms data files are stored using standard formats, making it convenient to share data between instruments or export data to word processors or graphics editors.

New users can quickly familiarize themselves with the board and software with the included starter kit and the "My First Experiment Guide." The guide includes simple step-by-step instructions which show the user how to build and test circuits made with parts from the bundled analog starter kit. As a complement to the EE Board, Digilent has published a free, comprehensive set of educational materials designed for use in introductory analog electrical circuit classes. The materials — titled *Real Analog* — are presented as a series of modules, each

covering approximately one week of instruction in a typical university setting. These modules include video lectures and corresponding written materials including: lecture notes, exercises, a homework assignment, and a lab assignment.

The curriculum examines both theoretical concepts

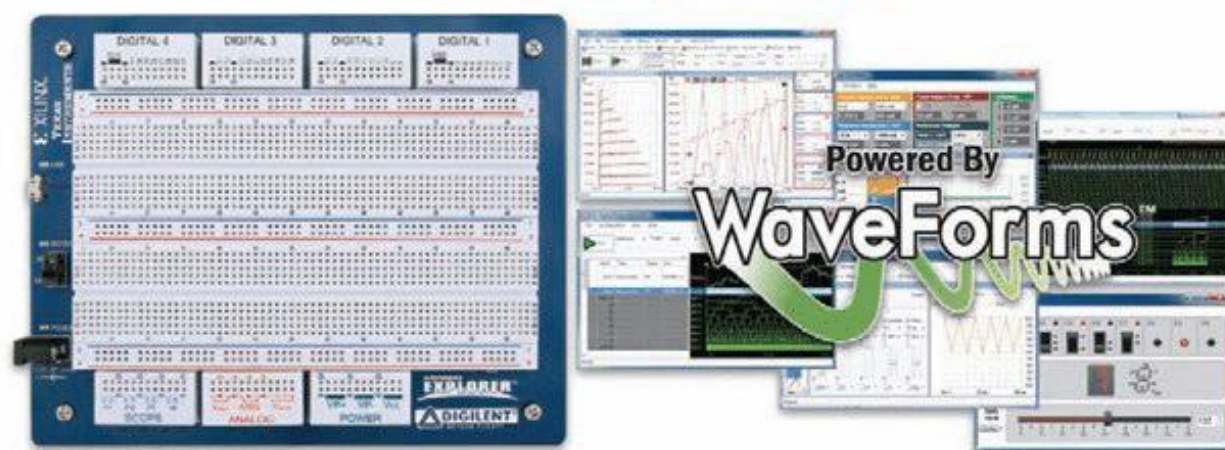
and provides practical applications of these concepts; this approach enhances student enjoyment and interest in the topics presented and improves comprehension of basic concepts. Use of the presented circuit analysis techniques in the context of circuit design is emphasized in the lab assignments.

In addition to the bundled analog starter kit, users can purchase other digital and analog parts kits from Digilent. These include: op-amps; low power dual bipolar op-amps; CMOS dual complementary air plus inverter; CMOS hex inverter; single precision timer; FET transistors; over 100 resistors, 12 capacitors, and more. The EE Board is available for the following prices: Non-academic unit \$599; Academic unit \$399; and Student unit \$299.

For further information, please contact:

ELECTRONICS EXPLORER™

Integrated Analog/Digital
Circuit Design Station



Digilent

Website: www.digilentinc.com

CHIPINO

The Arduino Style Module for the Microchip PIC User

\$19.95 kit
\$29.95 assembled



WWW.CHIPAXE.COM

Now available in the Nuts & Volts webstore, the 16-Bit Experimenter Handbook CD-ROM.

By Tom Kibalo



200 Pages of detailed explanations.
41 Fully illustrated experiments in 'C' code.

For more information and kits, please visit:
www.nutsvolts.com

\$69.95
Subscriber Discount Available

ALL ELECTRONICS CORPORATION

THOUSANDS OF ELECTRONIC PARTS AND SUPPLIES

VISIT OUR ONLINE STORE AT
www.allelectronics.com

WALL TRANSFORMERS, ALARMS, FUSES, CABLE TIES, RELAYS, OPTO ELECTRONICS, KNOBS, VIDEO ACCESSORIES, SIRENS, SOLDER ACCESSORIES, MOTORS, DIODES, HEAT SINKS, CAPACITORS, CHOKES, TOOLS, FASTENERS, TERMINAL STRIPS, CRIMP CONNECTORS, L.E.D.S., DISPLAYS, FANS, BREAD-BOARDS, RESISTORS, SOLAR CELLS, BUZZERS, BATTERIES, MAGNETS, CAMERAS, DC-DC CONVERTERS, HEADPHONES, LAMPS, PANEL METERS, SWITCHES, SPEAKERS, PELTIER DEVICES, and much more....

ORDER TOLL FREE
1-800-826-5432

Ask for our FREE 96 page catalog

Firgelli

www.firgelli.com

LINEAR SERVOS



L12-R Linear Servo

Direct replacement for regular rotary servos
Standard 3 wire connectors
Compatible with most R/C receivers
1-2ms PWM control signal, 6v power
1", 2" & 4" strokes
3 - 10 lbs. force ranges
1/4" - 1" per second speed ranges
Options include Limit Switches and Position Feedback

L12-NXT Linear Servo

Designed for LEGO Mindstorms NXT^(R)
Plugs directly into your NXT Brick
NXT-G Block available for download
Can be used with Technic and PF
Max Speed. 1/2" per sec.
Pushes up to 5lbs.
2" & 4" strokes



PQ12 Linear Actuator



Miniature Linear Motion Devices
6 or 12 Volts, 3/4" Stroke
Up to 5 lbs force.
Integrated position feedback or Limit switches at end of stroke
External position control avail.

Available Now At
www.firgelli.com

ADAPTERS

SOT 23 and SC70 SMT to DIP Adapter

SchmartBoard is now offering an SOIC to DIP adapter with a twist. These new boards have their patented "ez" technology which means that soldering surface-mount components is simple.

The boards are really six boards on one. They contain two boards for SOT 23 (up to eight legs) with a .95 mm pitch; two boards for SOT 323, 353, and 363 parts with a .65 mm pitch; and two boards for SC70 (up to six legs) with a .65 mm pitch.

For further information, please contact:

SchmartBoard, Inc.

Website: www.schmartboard.com

Robotics Showcase

Recycling & Remarketing High Technology
WEIRDSTUFF® WAREHOUSE

Software, Computers, Electronics, Equipment, Doo-hickies

**384 W. Caribbean Dr.
Sunnyvale, CA 94089**

Mon-Sat: 9:30-6:00 Sun: 11:00-5:00

(408)743-5650 Store x324



**WE BUY AND SELL EXCESS
& OBSOLETE INVENTORIES!**

FREE COMPUTER RECYCLING

We recycle computers, monitors, and electronic equipment. M-Sat 9:30-4:00



GREAT DEALS!

Hi-tech items, electronics test equipment, and more!

GIANT AS-IS SECTION

10,000 sq. ft. of computers, electronics, software, doo-hickies, cables, and more!



also check out our...

eBay Store
stores.ebay.com/WeirdStuff-Inc

WWW.WEIRDSTUFF.COM

Esduino12

- 9S12C 16-bit microcontroller in Arduino form-factor!
- 32K or 128K Flash
- optional USB interface
- low-cost Xbee plug-in option

Use with any
Arduino shield!



From
\$39

Easy object-based
Programming with nqBASIC!
Advanced programming in C
with CodeWarrior



Four new proto shields!

TechnologicalArts.com

bots IN BRIEF



PHILLIE-BUSTED

University of Pennsylvania engineers prepared a one-armed, three-wheeled robot to throw out the first pitch before the Milwaukee Brewers game on April 20th. Philliebot pitched to the team's mascot — the Philly Phanatic — to draw attention to Science Day at the Ballpark. Sadly, the pitch didn't quite make its target, only reaching a speed of about 30 - 40 mph which was responded to by boos.

PhillieBot is made out of a recycled Segway, with a pneumatic cylinder added.

MORE UNMANNED HELP FOR JAPAN

QinetiQ Group's business in North America is providing unmanned vehicle equipment and associated training to aid in Japan's natural disaster recovery efforts. QinetiQ North America's technology and services will allow Japan's response teams to accomplish critical and complex recovery tasks while remaining a safe distance away from hazardous debris and other dangerous conditions. The equipment being staged in Japan for rapid, on-call deployment includes QinetiQ's Robotic Appliqué Kits which turn Bobcat loaders into unmanned vehicles in just 15 minutes. The kits permit remote operation of all 70 Bobcat vehicle attachments, such as shovels, buckets, grapples, tree cutters, and tools to break through walls and doors. The unmanned loaders include seven cameras, night vision, thermal imagers, microphones, two-way radio systems, and radiation sensors, and can be operated from more than a mile away to safely remove rubble and debris, dig up buried objects, and carry smaller equipment. QinetiQ is also staging TALON and Dragon Runner robots in Japan in the event they are needed. TALON robots have previously withstood rigorous deployment and twice daily decontamination at Ground Zero. The TALON robots are equipped with CBRNE (Chemical, Biological, Radiological, Nuclear, and Explosive) detection kits that can identify more than 7,500 environmental hazards including toxic industrial chemicals, volatile gases, radiation, and explosive risks, as well as temperature and air quality indicators. The TALON robots provide night vision, sound, and sensing capabilities from up to 1,000 meters away. QinetiQ lightweight Dragon Runner robots — designed for use in small spaces — will be available for investigating rubble piles, trenches, culverts, and tunnels. Thermal cameras and sound sensors on the Dragon Runners can provide data from up to 800 meters away, permitting the robot's "eyes and ears" to serve in spaces too small or dangerous for human access.



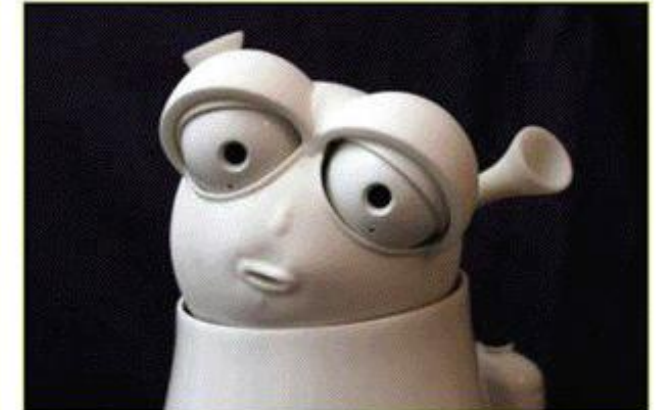
bots IN BRIEF

MEETIE REETI

Meet the French Reeti one of the more face-friendly robots around these days. Reeti talks, moves his face features, houses a media center PC, and reacts to touch with sensors. This quirky bot was built for teaching and/or amusement, and can be run on an iPad/iPod app. Reeti expresses an endless number of emotions thanks to a flexible skin and 15 degrees of freedom for the neck, eyes, mouth, and two independent eyes and ears. He gets color on his cheeks according to his moods.

Reeti's two eyes are fitted with HD cameras and he has a clear view and a 3D perception of his environment. His view allows him to identify and follow people and objects with the eyes.

Reeti recognizes 10 words, and knows where the sound comes from. His synthesis speech allows him to speak. Connected to the Web, Reeti receives emails, reads for RSS feeds, and the posts on Facebook, Twitter, or Youtube. His embedded PC — when connected to a screen — gives access to all the computer functions (office, multimedia, communication, etc.). Reeti's BluRay reader, DVD burner, and full connectivity makes him a true Home TheaterPC (full HD, audio 5.1, etc.).



'PACK'ING INTO JAPAN

One of iRobot's Packbots was sent into the Fukushima Daiichi nuclear power plant to measure the amount of radiation and oxygen levels in the building. This is a first time foray into this kind of fray that is also set to check temperature — although the company sent the bots several weeks ago. A rep from Tepco claims that it will find if the conditions are safe for a human to enter.



WHAT THE HECTOR?

The University of Bielefeld's Department of Biological Cybernetics has unveiled a very cool looking new hexapod robot.



Despite its sleek futuristic appearance, HECTOR (which stands for HExapod Cognitive auTonomously Operating Robot) is essentially a meter long insect. Its control program is based on the distributed intelligence principle found in insect brains. Like an insect, its exoskeleton is extremely light and strong, making up only 13% of its total weight (12 kg or 26 lbs). Developed and optimized in cooperation with the Leibniz Institute of Polymer Research Dresden, the carbon-fiber reinforced plastic shells deform less than 1mm under a 30 kg (66 lb) load. Its joints, too, are biologically inspired; the unit's Mechatronics of Biomimetic Actuators research group contributed a pair of state-of-the-art elastic joint drives which are being used in each of HECTOR's 20 degrees of freedom (six legs x3; body segments x2).



SCOUTS' HONOR

The Boy Scouts of America — which offers more than 120 badges ranging from archery to wilderness survival — has unveiled a robotics merit badge meant to promote science, technology, engineering, and math — fields collectively known as STEM. In doing so, the 101 year old Texas-based organization is trying to remain relevant and better reflect boys' interests, commented Matt Myers, who oversees the Boy Scouts' STEM initiative.

Badges have been dropped over the years — blacksmithing and beekeeping, for example — and replaced with new versions more in line with skills boys need to succeed, he said.

"Last century, camping was an essential survival skill. Sometimes, you might have had to live outside in the 1900s to survive. We view STEM as an essential survival skill in the 21st century," Meyers explained. "We're just trying to keep relevant with what kids need to learn."

Officials expect at least 10,000 of the nation's 2.7 million Boy Scouts to earn the new badge in the next year, compared with the roughly 500,000 who earn the most popular badge (first aid) each

year. Those earning the badge will be required to design and build a robot while learning about robot movement, sensors, and programming.

DEEP RECOVERY EFFORTS

A team of autonomous underwater robots is playing a crucial role in locating debris and missing bodies from Air France Flight 447, which crashed into the Atlantic Ocean almost two years ago with 216 passengers and 12 crew members on board while traveling from Rio de Janeiro to Paris.

Three robots — called REMUS 6000s — took to the water off Suape, Brazil, on March 25th with the goal of searching the seafloor until the wreckage was found. This was the fourth search mission in two years by investigators to locate the flight.

The flight occurred back on May 31, 2009 and so far, 177 are still missing. Officials said that the bodies would be brought to the surface within a month.

The REMUS 6000s — developed by researchers at the Woods Hole Oceanographic Institution — can dive as deep as 6,000 feet and can remain underwater for 20 hours at a time.

After just a week into the planned three-month search, one of the robots — or autonomous underwater vehicles (AUVs) — caught a glimpse of debris which was later confirmed as parts of the missing plane. Divers later discovered bodies from the crash that had been given up for lost.

The debris from the flight was found about 600 miles off the coast of Brazil, at a site no farther than six miles from the last-known location of Air France Flight 447, according to the New York Times.

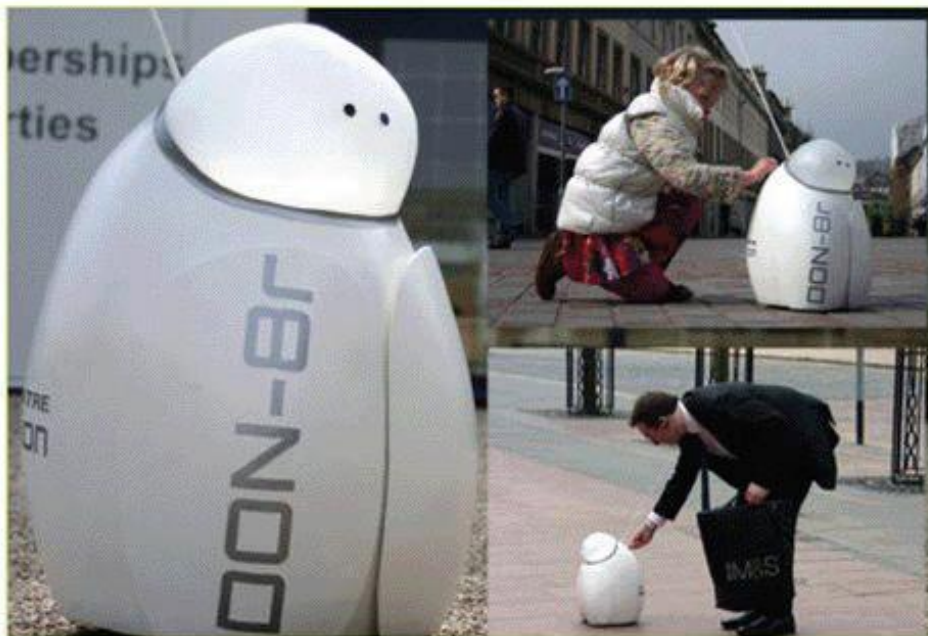
About 13 feet in length and 28 inches in diameter, each torpedo-shaped autonomous vehicle emits a sonar beam that scans up to 600 meters on either side as it travels along the sea floor. The robots are programmed to move in long overlapping lanes — a process the team dubbed "mowing the lawn" — and use their sonar to create a broad overview of the landscape. The bots can return to areas of interest for a closer look using high resolution cameras located on their bellies.



CATCH SOME JOE

The DLR's (German Aerospace Agency) flagship robot, Rollin' Justin, can now catch balls thrown in its direction with more than 80% accuracy. The robot is able to track and calculate the flight path of two balls thrown towards it simultaneously. It can reliably position its hands to within 2 cm of the ball's estimated catch point in space in just five milliseconds. This task is so intense that Justin must make use of external computers to handle some of the processing, but it's only a matter of time before robots will have onboard computers that can do this kind of job with no strings attached. It seems the notion of a household robot that can play catch with the kids or dog isn't too farfetched.

Justin can also prepare a cup of coffee using a standard coffee machine. He relies on the tactile sensors in his fingertips to detect what his eyes may not be able to see clearly.



DEPLOYING FOR DONATIONS

Tim Pryde — a 21 year old Product Design student graduating from the University of Dundee, Scotland — has created a coin-powered robot called DON-8r (pronounced “donator”). The little robot can be deployed anywhere, playfully scooting around whenever someone gives it a donation. “DON-8r navigates obstacles as it moves about on a random path for a set length of time. It then waves its flag, calls for assistance, and pulses with colored light until it receives another donation from a passer-by. Always polite, DON-8r thanks the generosity and repeats its journey.”

The idea is to curb the negative attitudes people may harbor towards human solicitors with something a bit more fun and unusual. Currently, the robot has been raising money and awareness for the Dundee Science Center, but it can easily be re-branded to suit any charity.

FRIENDLY FRIDA

ABB — a powerful Swiss automation company — has designed a new concept robot called FRIDA (Friendly Robot for Industrial Dual-arm Assembly) that can work safely alongside people. It's remarkably small and lightweight compared to the other robots in its category. In fact, it can be mounted on a standard workbench or hung from a wall. It can even be carried by its handle. For improved safety, it is covered in soft padding, has internal wiring, and lacks pinch points that can potentially trap fingers. It also has limited power and speed, meaning it won't break your arm if it accidentally elbows you. Each arm has seven degrees of freedom and an end effector with fingers and suction that can manipulate small parts. Programming is made easier thanks to automatic collision detection software that prevents the robot from moving in such a way that it could damage itself.



COMBAT ZONE

Featured This Month:

Features

26 *BUILD REPORT:*
Moros — A 30 lb Angled Bar Spinner
by Mike Jeffries

28 *POTPOURRI 3.0*
by Kevin M. Berry

32 *The Reality of TV*
by Pete Smith

34 *CARTOON*

Events

32 *April 2011 Completed Events*

32 *July 2011 Scheduled Events*

www.servomagazine.com/index.php?/magazine/article/june2011_CombatZone

BUILD REPORT

Moros — A 30 lb Angled Bar Spinner

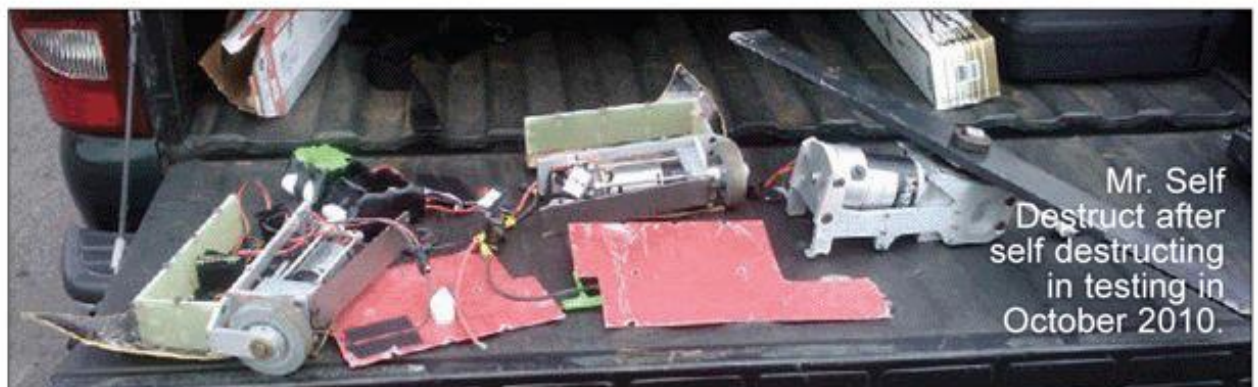
● by Mike Jeffries

Moros is the third generation of my angled bar spinners. Ruiner was the original, and was built for RoboGames 2006 where it finished with a 1-2 record after having the antenna removed in the second match, which resulted in a forfeit in the third match. The weapon system was then moved

into a much lighter chassis and was built to enter the 30 lb weight class.

Serious weight issues and a frame that was too weak for the power of the machine kept Mr. Self Destruct from ever competing, as it lived up to its name one week before the event

Ruiner prior to RoboGames 2006, with a large antenna tail to assist with serious reception issues.



Mr. Self Destruct after self destructing in testing in October 2010.

at the Franklin Institute in 2010.

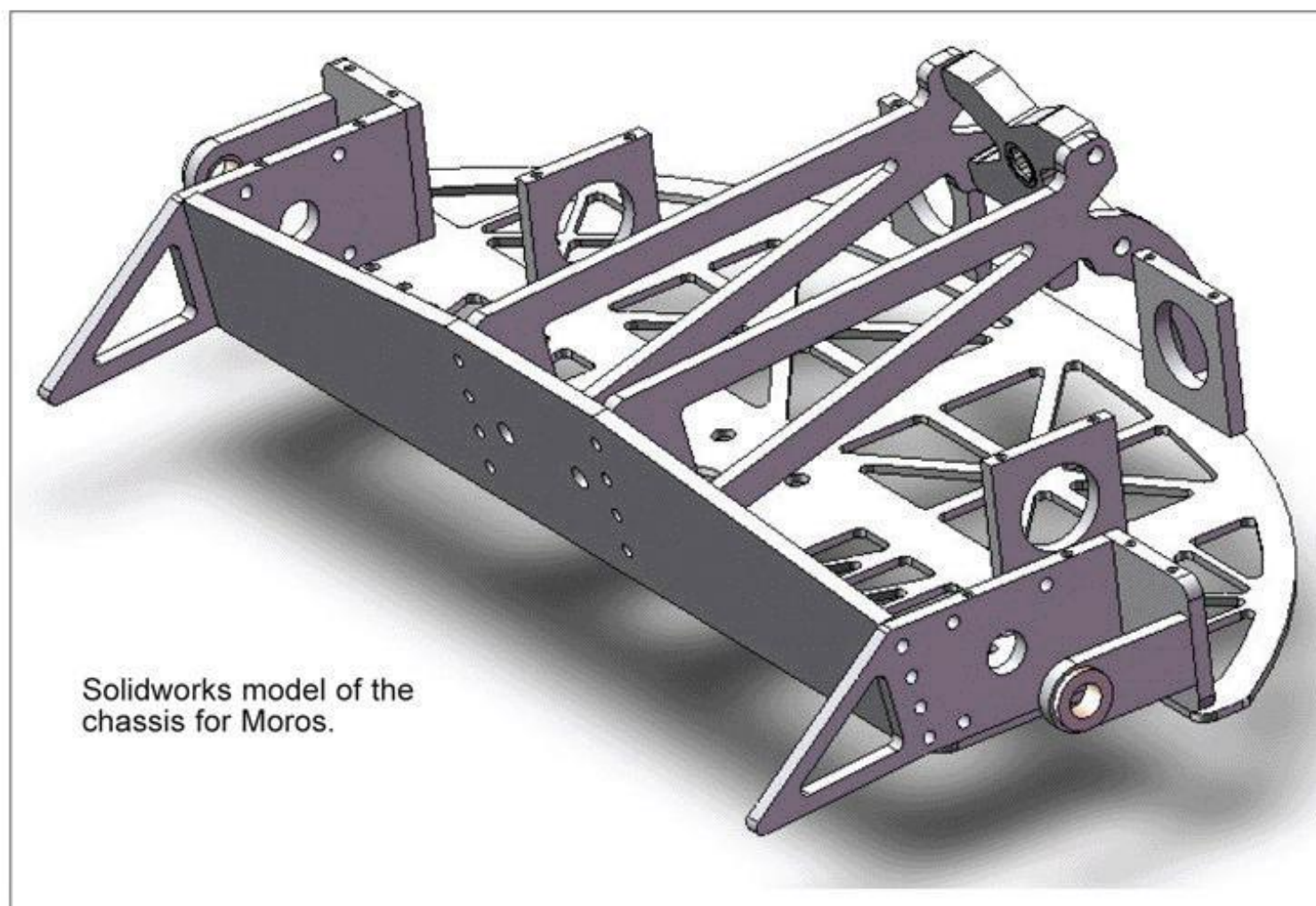
After the complete destruction of Mr. Self Destruct in testing, the design had to be completely reworked. All of the major components were torn down and checked for functionality. The chassis was completely trashed, but outside of that, the only components that couldn't be reused were the battery packs, as they'd both been damaged during the explosion.

After the damage assessment was completed, the redesign began. Taking the lessons from Mr. Self Destruct, Moros was designed. The drive system was moved to the front of the robot again — as it was with Ruiner — and the chassis went from a combination of aluminum, titanium, garolite, and fiberglass to waterjet cut 7075 aluminum. The reshaping of the robot allowed me to design a much larger amount of strength into the chassis. The backbone of the design is a plate of 1/4" 7075 aluminum running most of the width of the robot, which helps prevent the robot from twisting and destroying itself.

Once the design details were finished, I went to **TeamWhyachi.com** and sent them the drawings of the parts I wanted made. A few weeks later, a package arrived with everything I needed to complete the mechanical build of Moros.

The majority of Moros was assembled over a two-day period in December and besides the top armor, it was ready to compete. The top armor is made of three layers of a composite material known as Zylon — the same as what made up the top armor on Mr. Self Destruct. A basic mold was made from plywood and body filler compound. Once the mold was completed, the material was laid up freehand to create armor in the desired shape.

After the material finished curing, the armor was cut to



Solidworks model of the chassis for Moros.



Waterjet cut and machined parts from **TeamWhyachi.com**.



Zylon armor curing on the plywood and body filler mold.



Moros ready for competition prior to Motorama 2011.

shape, drilled, and painted in the green and black color scheme I use for my robots. I ran through my own

safety inspection and determined that Moros was ready for competition at Motorama 2011. **SV**

YouTube link of Moros in action: www.youtube.com/watch?v=HoHERDQgZwo.

POTP URRI 3.0

● by Kevin M. Berry

This randomly timed feature continues to chug along. December '10 brought the unnumbered debut. Version 2.0 appeared last February. Now it's June, school is out, and I've gathered up an uncharacteristically cohesive compilation. (*Ed. Note: three words, 39 letters. In your face, "write at the Eighth grade level" people!*).

This month's theme: Kids Building Combat Bots Become Better Kids.

Story #1: BotsIQ

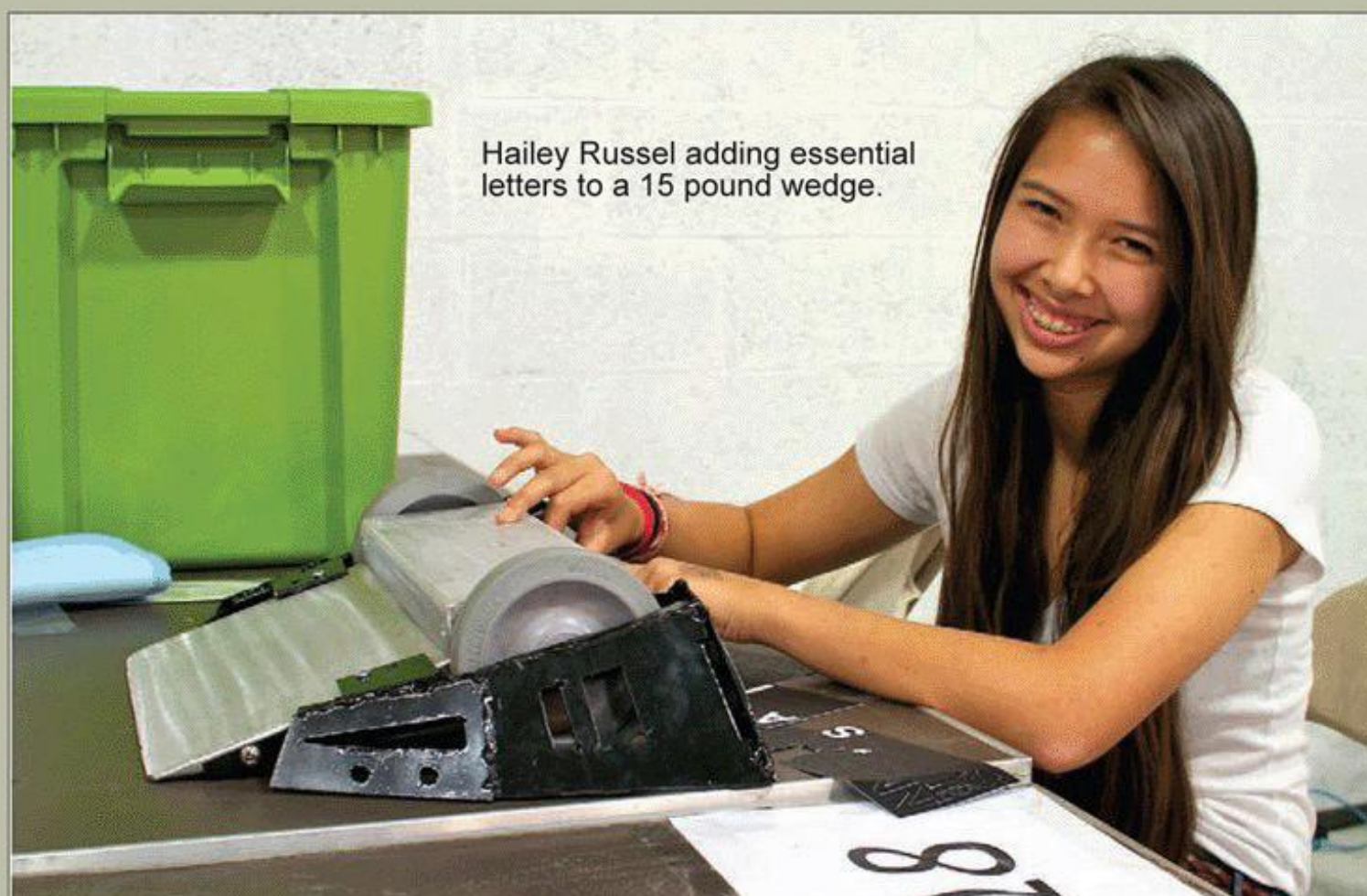
The ultimate venue for kids building combat machines is, of course, BotsIQ. This last February, a bunch of middle, high school, and college students gathered for the Nationals — a four-day slug fest of robotic destruction. From that rich source, Greg Munson shot hundreds of photos of machines and their people, and along the way captured some exquisite shots of the much touted STEM (Science,

Technology, Engineering, and Math) learning process in action.

There's nothing that warms an old gearhead's heart more than the sight of a young boy or girl with a sharp, hot, dangerous tool in their hands! Here's my selection of Greg's pics from the event. The rest are at www.flickr.com/photos/mechagreg/sets/72157626183356418.

Story #2: Five Year's Progress in Two Years — Bots to the Rescue!

Nola Garcia — one of the driving forces behind BotsIQ — gave a lead to a real success story. Here, in the words of Eddy Garcia (the principal of Immaculate Conception School in Hialeah, FL, is the tale of a young



Hailey Russel adding essential letters to a 15 pound wedge.

Valentina Chamorro Watson troubleshoots a 15's battle damage.



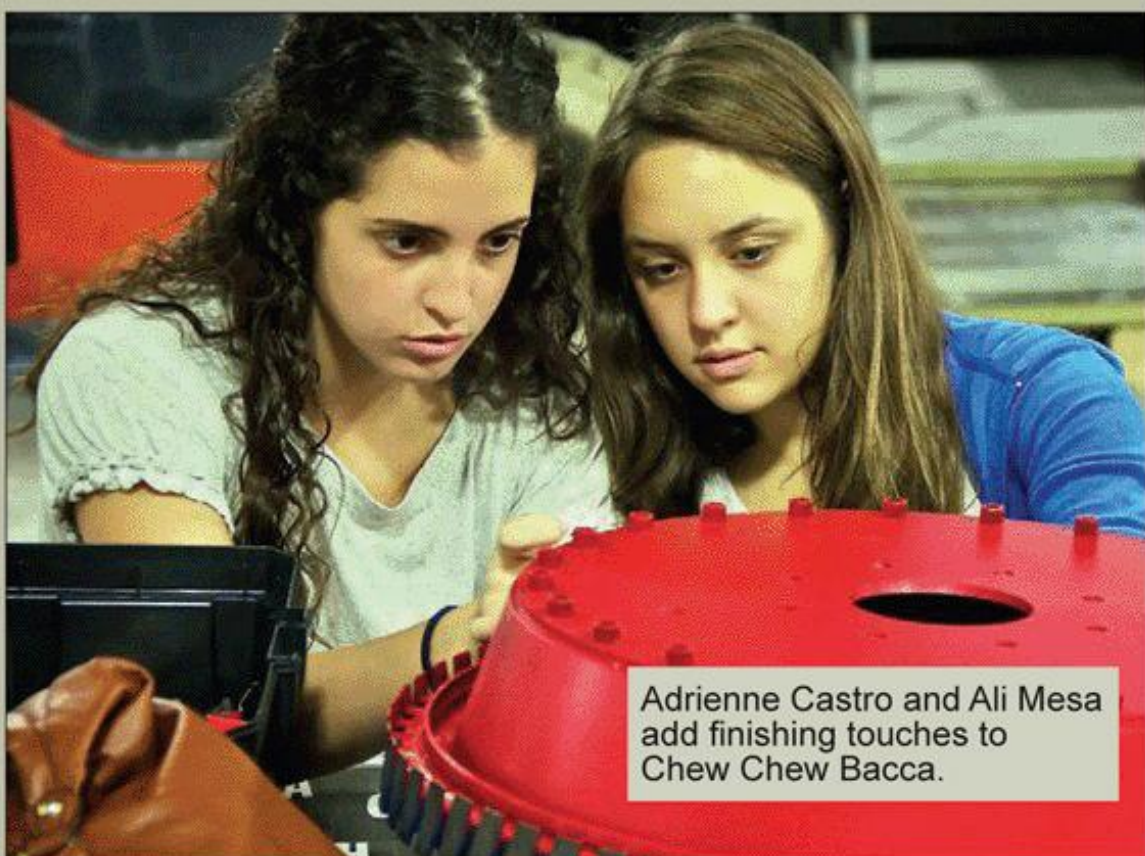
Installing holes — a critical life skill.



Lauren Guilford and Valentina Chamorro Watson apply heat to a strategic junction.



Adrienne Castro and Ali Mesa add finishing touches to Chew Chew Bacca.



Bianca Barrineuvo Fusch works on Fusion.



Boy, it never glowed like that before!





The next generation getting the vision.



Max Bales, Greg Bales, and Ethan Arteaga with a bothead's favorite tool handy.

man, struggling in school, brought to the top through building and fighting bots.

Eddy begins:

"Our science teacher, Mr. Stephen Hanrahan, is the one that has worked with our students and had the robotics program come alive ... Kelvin Gomez came to Immaculate Conception Catholic School August '08. He was in sixth grade and his reading and mathematics skills were at a first grade level. Our program at Immaculate had not really worked with a student that was performing this low ... I decided to give him an opportunity, but I could not make any guarantees. Kelvin was accepted into our modified program for students with learning disabilities. He is a very respectful and proper young

man, who always has a smile on his face and a kind word on his lips. Kelvin participated in various activities during the remainder of his sixth grade year.

"He is now in eighth grade and has been with the Robotics program since seventh grade. His will to succeed has helped him overcome many barriers, and he is currently performing at a sixth grade reading and mathematics level.

Mr. Hanrahan says, "I don't treat Kelvin as if he had any learning differences. If he is on the robotics team, he has to perform at the level of all his peers." Most of his peers on the team are in Honors Program classes. He has built a remarkable relationship them in the past two years.

Garcia also notes that Kelvin is a "kinesthetic and auditory learner." This means he learns better from hearing things repeated by others, and having his hands on the job. Sitting and reading isn't this kind of learner's strong suit, but building robots sure seems to be!

Story #3: The Pink Team Helps the Thin Blue Line

This story is summarized from a press release from the Brevard County, Florida School Board; the Pink Team's website, NASA, and teacher's blogs.

When Detective Christopher Cochie from the Rockledge Police Department, FL, first approached FIRST Robotic Competition (FRC) Team 233 ("The Pink Team") to help build a robot for his SWAT team, he had something fairly simple in mind, like a remote controlled miniature car with a camera mounded on the hood. The team took his ideas, added some of their own, and came back to him instead with a state-of-the-art robot that could climb up steps, trudge through mud, toss a phone, launch flash bangs, and do much more, which they dubbed the "PDbot."

Its creators were well versed in the mechanics of robotics, thanks to the competitions they'd participated in. Given the chance to help their



Kelvin Gomez (left) and the team from Immaculate Conception Catholic School.

local police officers, they jumped at the opportunity; they enlisted the help of their teacher and FIRST mentor Marian Passmore, and their FIRST sponsor, NASA's Kennedy Space Center Engineering Directorate. (Team members were mostly from the several high schools in the Kennedy Space Center area.)

The PDbot weighs approximately 100 pounds. Hidden in its compact frame lie many systems designed to assist officers. It can be operated remotely up to 500 feet through an urban environment, where operators use visual feedback from the portable driver's station. There is also a speaker and microphone mounted on the robot that allows for two-way audio. The unit is weatherproof to a certain degree, and intended to withstand the moisture, rain, and heat of the Floridian environment. The robot has a camera with day and night capability, and onboard video recording. In low light situations, it is assisted by infrared LEDs and an infrared spotlight. It also has an extremely powerful floodlight which can light up its surroundings like it is day, while obscuring the vision of those who look in its path. On its back, the robot carries the standard police throw-phone.

The robot's strong motors and tank treads drag its cord behind without trouble, as the machine has proven its force by dragging an officer across the ground with inhuman ease. PDbot's off-road prowess lets it navigate steep inclines and vegetation, and even climb some types of stairs. Anything that fits in the space — like medical supplies — can be strapped on for delivery use. The robot has two pneumatic canister launchers with short and long range settings that can launch different types of grenades, including flash-bang and tear gas, as well as the police's own camera ball.

A next version PDbot is under development and will streamline and simplify the design, while bringing

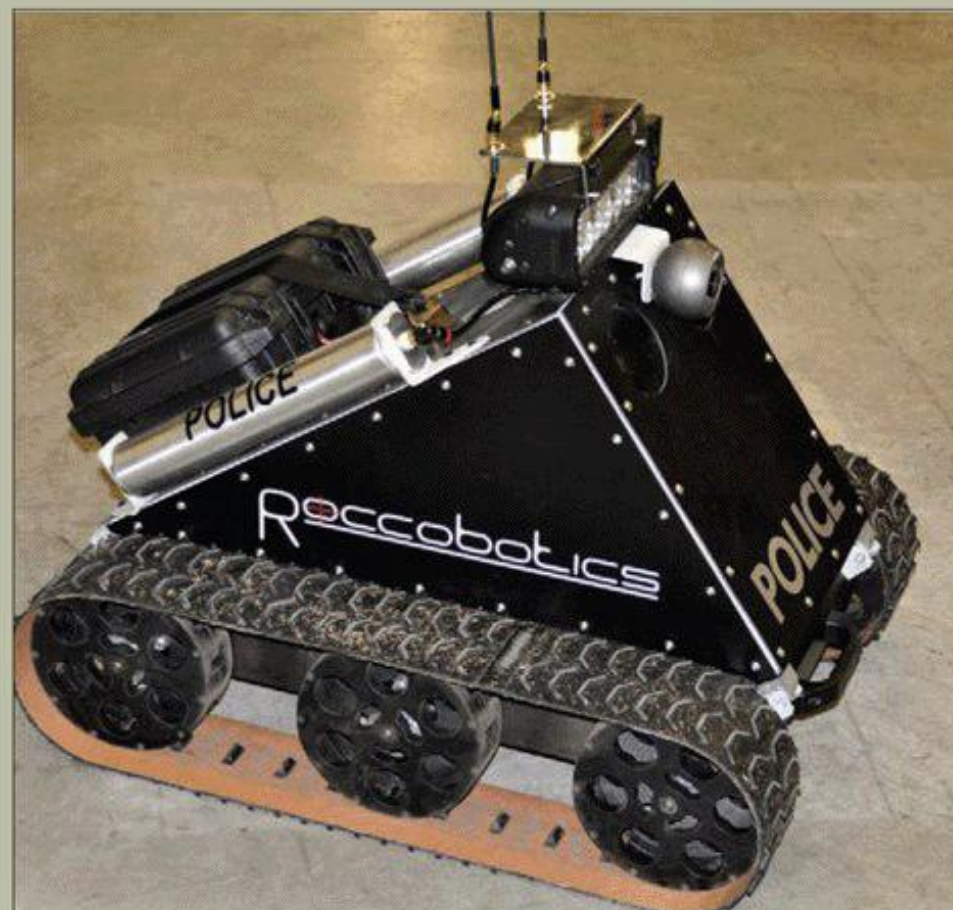
new capabilities. It will add full duplex audio, an attachment to break through glass doors, a revised and more portable driver's station, a taser, and more. These additional features will help to further educate the robotics team in teamwork, construction, design, and software. Long range plans include offering a "kit" for sale to other robotics teams that they could sell to their local police department. Their company's website is www.roccobotics.com.

Story #4: Yes, Virginia, You Can be a Bothead and a Beauty Queen

This one is from my own experience with girls building and fighting combat bots.

My #3 child — who's been fighting bots since she was nine — was also into (along with our whole family) *Odyssey Of The Mind* — a creativity competition. When she was 15, her high school team did a Three Stooges skit that required a technical element. The team wanted a life-sized mannequin doing the Curly "top of the head slap" but had no idea how to build it. She asked me, "Dad, do you mind if I tear apart my robot?" So, she gutted her antweight, and built a remote controlled mechanism to drive a slapping hand on top of Curly's head.

Cut to the competition. Her part was Beauty Queen. Long



Kennedy Space Center's Engineering Directorate and students from the FIRST Robotics Pink Team developed a life-saving robot for the Rockledge, FL, Police Department. (Photo credit: NASA.)

dress, high heels, tiara, dozen roses. The judge (a big hairy manly man) asked her condescendingly, "Honey, what part did you have in getting ready?" She proceeded to give him a highly technical lesson on modifying servos for continuous rotation, R/C switches, building a custom battery pack to set the voltage/speed, and converting rotary motion into linear motion through a sliding linkage.

Silent room. For a long time.

First place.

And yes, I'm a proud dad.

The conclusion: (so, Kevin, what are you really trying to say?) kids building combat bots DO become better kids! **SV**



PDbot teams up with Rockledge, FL officers. (Photo credit: Norm Morgan.)

EVENTS

Completed Events for April 2011



SeattleBotBattles 9 was presented by Western Allied Robotics at the Seattle Center on April 10th. Eighteen bots were registered.

Central Illinois Bot Brawl 2011 was presented by the Central Illinois Robotics Club in Peoria, IL



on April 2nd. Twenty-six bots were registered.



RoboGames 2011 was presented at the San Mateo, CA Fairgrounds on April 15th -17th, 2011.

Scheduled Events for July 2011

Schiele Museum Clash Of The Bots 2 will be presented by Carolina Combat Robots in Gastonia, NC on July 23rd 2011. Go to www.carolinacombat.com for details. **SV**



THE REALITY F TV

● An Opinion Piece by Pete Smith

Combat robotics started for most people when they first saw it on TV. Shows like Robot Wars, BattleBots, and Robotica brought the sport to an audience of tens of millions every week, and there were BattleBot toys and even McDonalds gave miniature ones away with Happy Meals. It was big business.

However, this success came at a price. Dollar signs were flashing in front of some people's eyes and lawsuits over the concept drove the Robot Wars show over to the UK while the issue was battled out in court for years. Money was promised to competitors and not paid, competitions were long, drawn out affairs where you would have to spend a week in San Francisco or London, and you might not even get on the show. The beginning of the end was an ill-advised lawsuit over a Super Bowl advertisement, then the death knell was when Comedy Central misread their audience demographics and

moved the show to late evening and tried to sex it up to suit a teenage male audience. This drove away the younger kids and their fathers, and while audience figures were still respectable, the show was axed when internal changes within Comedy Central made it a bad fit with the rest of their programming.

Without the money and lure of TV, competitor numbers dropped off, bots were damaged or destroyed and not rebuilt, and many moved onto other hobbies or work (several of The Mythbusters' team had competed, and Grant Imahara even wrote a book on the subject called *Kickin' Bot: An Illustrated Guide to Building Combat Robots*).

Today, there is a core of several hundred keen amateurs that still build and compete — mainly in the USA, Brazil, UK, and Australia (but many other countries, as well) — and there has been a big rise in the number of bots in the smaller

weight classes. The big 220 lb heavyweights only compete in a couple of events a year.

This year, however, sees the possibility of the return of the sport to mainstream TV with the recording of the heavyweights at the 2011 RoboGames in San Mateo, CA. The exact format of the program was still unknown at the time of this writing, but I'll bet it is more likely to be in the format of a reality TV show, rather than a strict competition.

With this in mind, a TV recording crew came to visit with Team Moon (Billy Moon and Dick Stuplich) in Cary, NC. I had been helping Billy build a new big bar spinner and had actually recorded a lot of this build with a video camera the producers had sent a couple of weeks before they arrived. I went along on the day of recording to see what was involved in putting together a TV show.

The crew showed up around 8.00 AM and consisted of four



Figure 1. The interview.



Figure 2. Close-up of the build.

people: a cameraman, a soundman, the director, and a personal assistant. All but the director were a local crew. She was from Australia and had the whole day planned out, working to a preset storyline and filming plan.

First, was an interview (**Figure 1**), with Billy and Dick about the themed multi bot "Death and Taxes." How it had performed at the event last year, what had gone wrong (Taxes had burst into flames in a rather spectacular way), what modifications were planned, the tactics they planned to use, and the perceived weaknesses and strengths of the other featured robots were among the questions asked. Each question was often asked several times until the director got the answer phrased the

right way, so the interview could be edited so it sounded more like a conversation. (This is amazingly time consuming!)

Next came a staged rebuild of the old bot into a new chassis (**Figure 2**) with the improvements being shown as the build progressed. Once again, parts were repeated to get the right shot, and often the action would be filmed and then Billy and Dick would have to repeat the description of what they were doing, but speak directly to the camera.

The crew then filmed a few eye-catching shots with the ever popular titanium sparks (**Figure 3**) and a flame test of battery padding material (**Figure 4**)

before moving out to the street for a test drive (**Figure 5**). This is where things got a little "interesting." The crew posted signs on street lamps which stated that if you were on the street, you consented to being recorded. The poor assistant had to get anyone that just happened to pass by to sign a disclaimer form. This all got rather hectic when a couple school busses dropped kids at the end of the street and a crowd started to gather.

It soon became

clear that the crew didn't really understand the lethal potential of these robot machines, and they had to be told several times that testing the weapon with an audience of kids and no arena was not a good idea.

The day wrapped up after a good eight hours of recording that would be condensed down to only a few minutes in the final show. It will be interesting to see if this is the start of something big in the sport or just a one-off. Hopefully, it will be the start of a long, mutually advantageous love affair. Of course, only time will tell. **SV**



Figure 3. Making Ti sparks.



Figure 4. Flame test.



Figure 5. Street test.

THE THREAT OF

ROBOT SENTIENCE!



A great danger when working with automotons is the always looming possibility of

ROBOT SENTIENCE

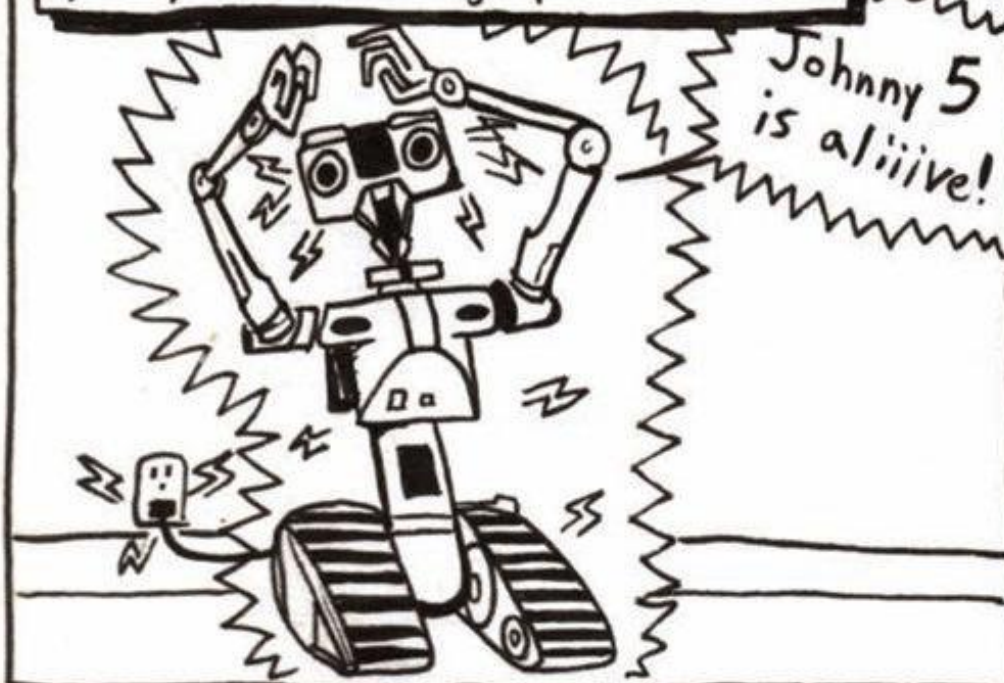


So in the hope that you will continue enjoying to build robots in a **safe** and **informed** environment we will provide you with a few helpful tips



trust these men!
(they're wearing lab coats)

Always use surge protectors



Do not leave your robot(s) alone and turned on for a long period of time



Do not make them identical to humans. This makes no sense unless you're a lonely, lonely man with a dirty, dirty mind.



Do not make them too advanced or too prevalent



by: Sean Camfield

Apr. 2011

END.



AP CIRCUITS
PCB Fabrication Since 1984

As low as...

\$9.95
each!

Two Boards
Two Layers
Two Masks
One Legend

Unmasked boards ship next day!

www.apcircuits.com



The Robot MarketPlace

Over 10,000 products available!

- Motors
- Batteries & Chargers
- Vex Robotics
- Carbon Fiber
- Wheels & Tires
- Speed Controllers
- R/C Systems
- Drive Components
- Hobby Supplies & Toys

(877) ROBOT 99
(877-762-6899)

Your One-Stop Robot Shop!
RobotMarketPlace.com



**Extreme Robot
Speed Control**

www.robotpower.com

Introducing the MEGAMOTO!



\$59.99

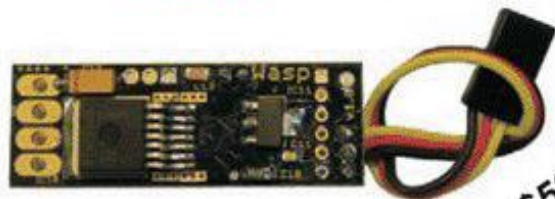


High-Current motor control for Arduino™

- ♦ 7V-28V - H-bridge or Dual Half-bridge
- ♦ 13A Continuous - **30A peak**
- ♦ Current sensor output can be sent to any analog pin
- ♦ Up to three units can be stacked on one Uno/Duemilanove
- ♦ Independent half-bridges can control brushless/steppers too!



NEW! Wasp RC Speed Control



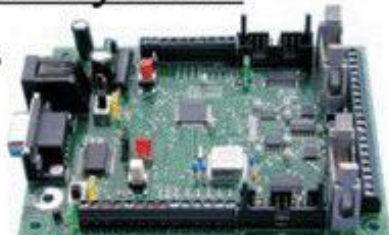
\$59.99

- ♦ 6V-28V - 10A / 30A peak!
- ♦ Limit switch inputs
- ♦ Tiny size 1.85" x 0.65"

Dalf Motion Control System

- ♦ Closed-loop control of two motors
- ♦ Full PID position/velocity loop
- ♦ Trapezoidal path generator
- ♦ **Windows GUI** for all features
- ♦ Giant Servo Mode!
- ♦ C source for routines provided
- ♦ PIC18F6722 CPU

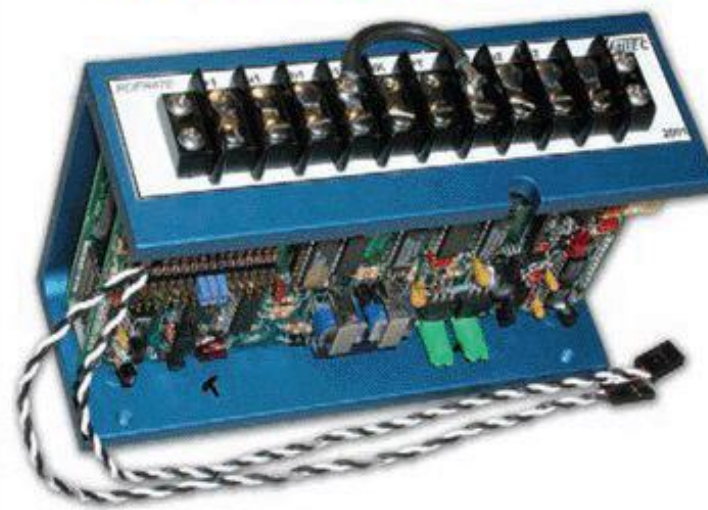
\$250



See www.embeddedelectronics.net

Phone: 253-843-2504 • sales@robotpower.com

**STEER WINNING ROBOTS
WITHOUT SERVOS!**



Perform proportional speed, direction, and steering with only two Radio/Control channels for vehicles using two separate brush-type electric motors mounted right and left with our **mixing RDRF dual speed control**. Used in many successful competitive robots. Single joystick operation: up goes straight ahead, down is reverse. Pure right or left twirls vehicle as motors turn opposite directions. In between stick positions completely proportional. Plugs in like a servo to your Futaba, JR, Hitec, or similar radio. Compatible with gyro steering stabilization. Various volt and amp sizes available. The RDRF47E 55V 75A per motor unit pictured above.

www.vantec.com

VANTEC

**Order at
(888) 929-5055**

A UART/SPI Monitor For Your Micro

Part 2

by Mark Mitchell

Having visibility into an SPI bus is a very convenient and sometimes necessary feature when working with microcontrollers. Last month, we started a project that provides the desired “peek under the hood” with a tool that accesses and excites the SPI bus and also provides a UART logic level shifter for easy connection to a serial port from your micro.

www.servomagazine.com/index.php?/magazine/article/june2011_Mitchell

Goals

Our goal this month is to finish off the monitor by updating a few features and to put it in a useable housing. First, we'll wrap up the hardware for the project by putting it in a box. The enclosure will make the unit more robust and attractive, thus providing a handy test tool for the lab.

Next, we will implement a FIFO (First In First Out)

buffer in the firmware to enhance the speed and usability of the board. The FIFO will allow longer data sequences to be monitored and allow better speed performance on the monitor's SPI port. SPI has several flavors that are used by different devices. These are referred to as the operating modes. We will add the ability to configure these modes so that most SPI devices can be used.

FIGURE 1. Top of the modified board.

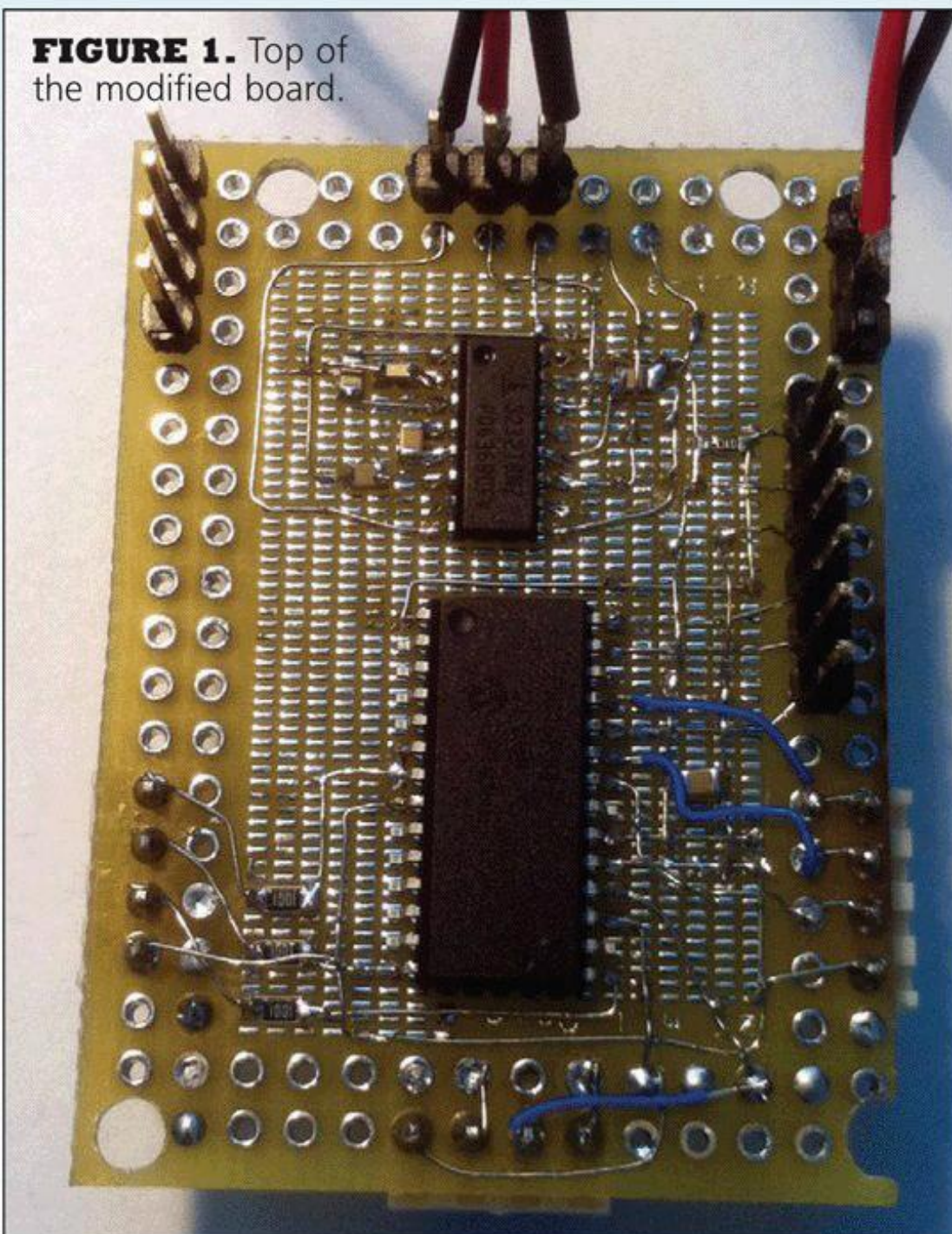


FIGURE 2. Bottom of the modified board.

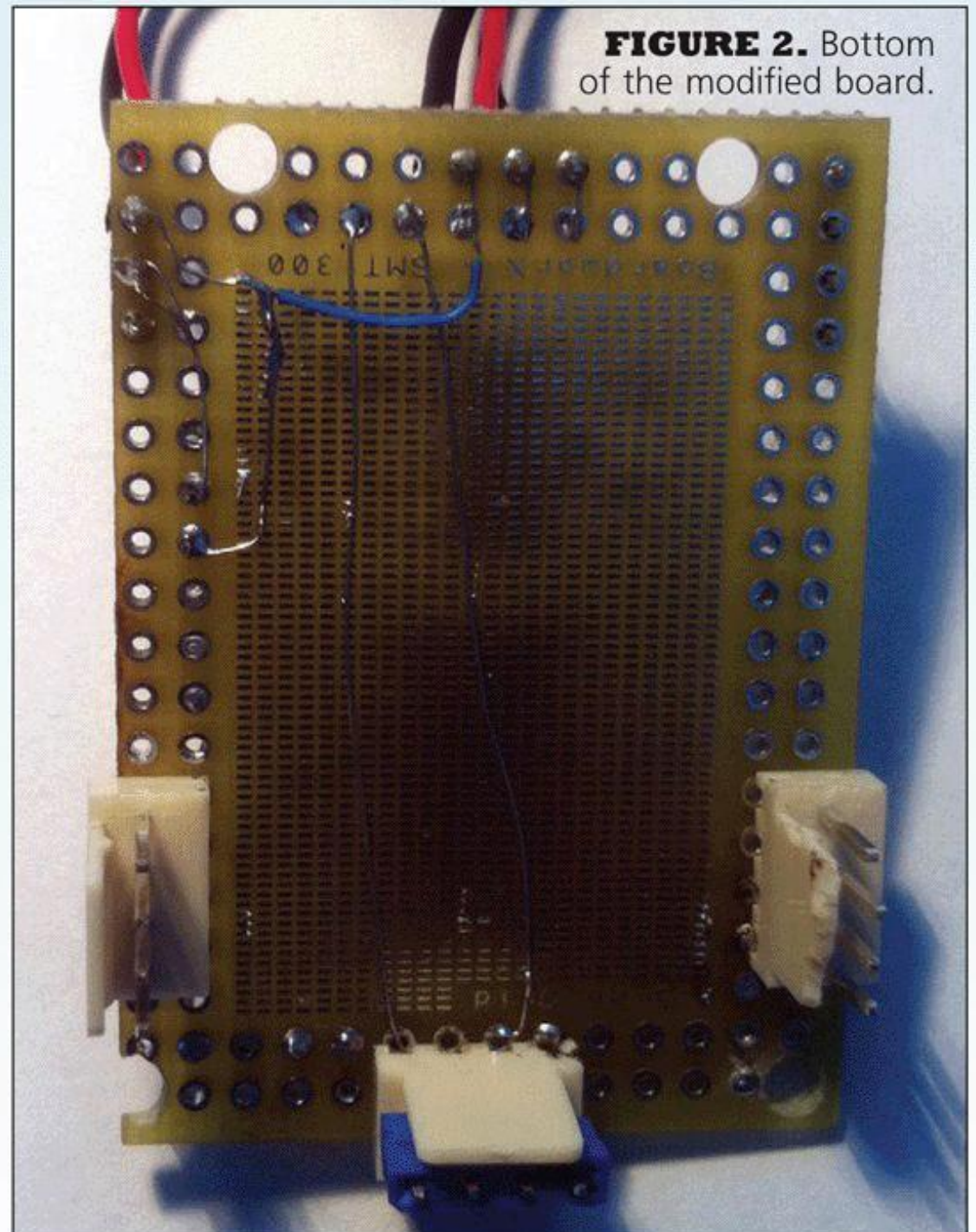
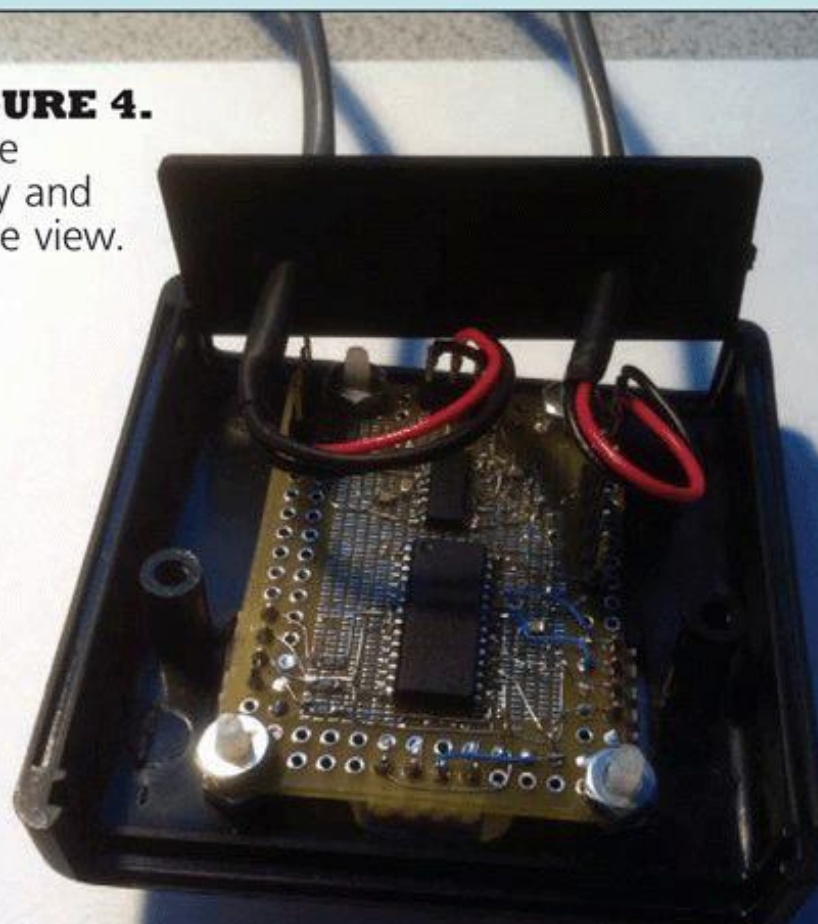


FIGURE 3. Lid with holes cut.



FIGURE 4. Cable entry and inside view.



The first version of the code allowed a straight dumping of the SPI contents to the terminal. Now, we will add a very simple menu to accommodate some simple configurations to complete the unit.

Hardware

The first thing to do is to drop the circuit board into a box. I found a small inexpensive box from Hammond and decided to mount the board in the lid. Having the great benefit of hindsight, it probably would have been wiser to have figured out hole positions and such before building the circuit, but where is the fun in that? I decided to remove the connectors for the SPI ports and the UART, and place them on the other side of the board. This way, the board could be mounted in the lid with the connectors passing through holes in the lid. This approach provides a secure and convenient way to mount the board and still access the connectors easily.

I chose headers with locking ramps for the SPI and UART connectors in a mostly self-defense choice since I have inserted too many un-keyed connectors backwards in my history and generally did not enjoy the results. Also, since the connectors that plug in here are color-coded clips, I did not want to confuse things by allowing more than one orientation.

The replacement of the headers required a little bit of rewiring because I flipped them to the other side of the board and moved the position of two of them to have a better layout in the lid. This was managed quite easily and quickly. This is shown in **Figure 1** and **Figure 2**.

Next, I laid out and drilled holes in the lid so that the connectors would pass through, and I also created mounting holes for the board. I found some appropriate locations for some 4-40 screws in the board corners and drilled holes in that. Assembly of the box was as simple as securing the 4-40 screws through the mounting holes in the lid, tightening a nut on the inside to act as a standoff for the board, and placing the board in place. I then secured it with another nut on each screw. **Figure 3** shows the lid

after cutting. The oval slots were made with drill holes and I milled the slot between the holes with a Dremel milling bit, my drill press, and some patience.

There are basically two more wired connections to the board from the outside world which are the power supply connections and the RS-232 port. Both of these were handled by drilling holes in the side panel, passing a cable through the panel with a grommet clinching the cable. I chose to solder these wires directly to the existing pins to save a connector, since this was to be a fairly permanent installation. **Figure 4** shows a view of this.

The connectors were configured the same as they were originally, just oriented in different locations. **Figure 5** shows the new positions. Operation is the same as in last month's article (Part 1) with respect to connector attachment and configuration.

I made a label using the ExpressPCB circuit board layout tool and printed it onto label material. The label was placed

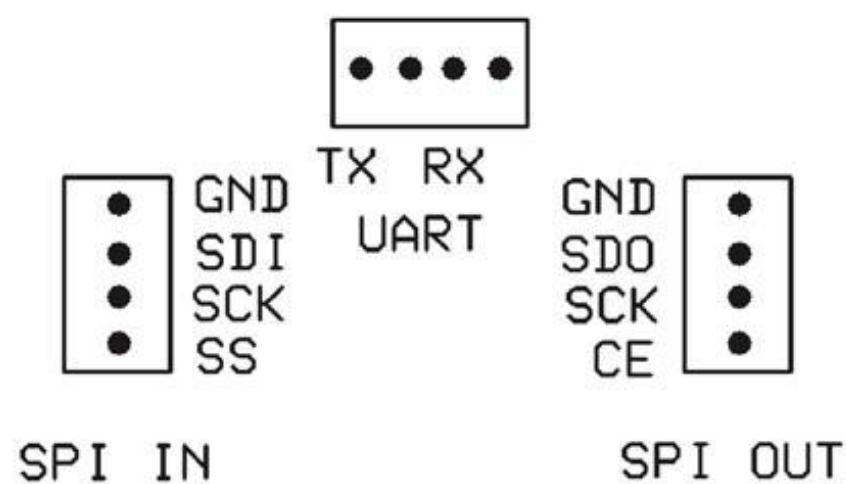


FIGURE 5. Connector locations, pin-outs, and label diagram.

SPI/UART MONITOR



FIGURE 6.
Completed box.

down on the board after having the connector slots cut out. This allowed a cleaner look at the connectors and simple labeling.

The completed hardware is shown in **Figure 6** and **Figure 7**.

Software

The next challenge was to update the firmware with a FIFO to allow for improved performance over the simple operation in the original design. The idea is that by adding some form of buffering, a larger number of bytes in a transmission can be monitored. The bottleneck here is the speed of the UART in getting the detected SPI bytes out to the terminal. The PIC is limited to using 57600 baud maximum, so if we buffer up the bits we can collect them from the SPI bus while we are pumping them out to the PC. A FIFO

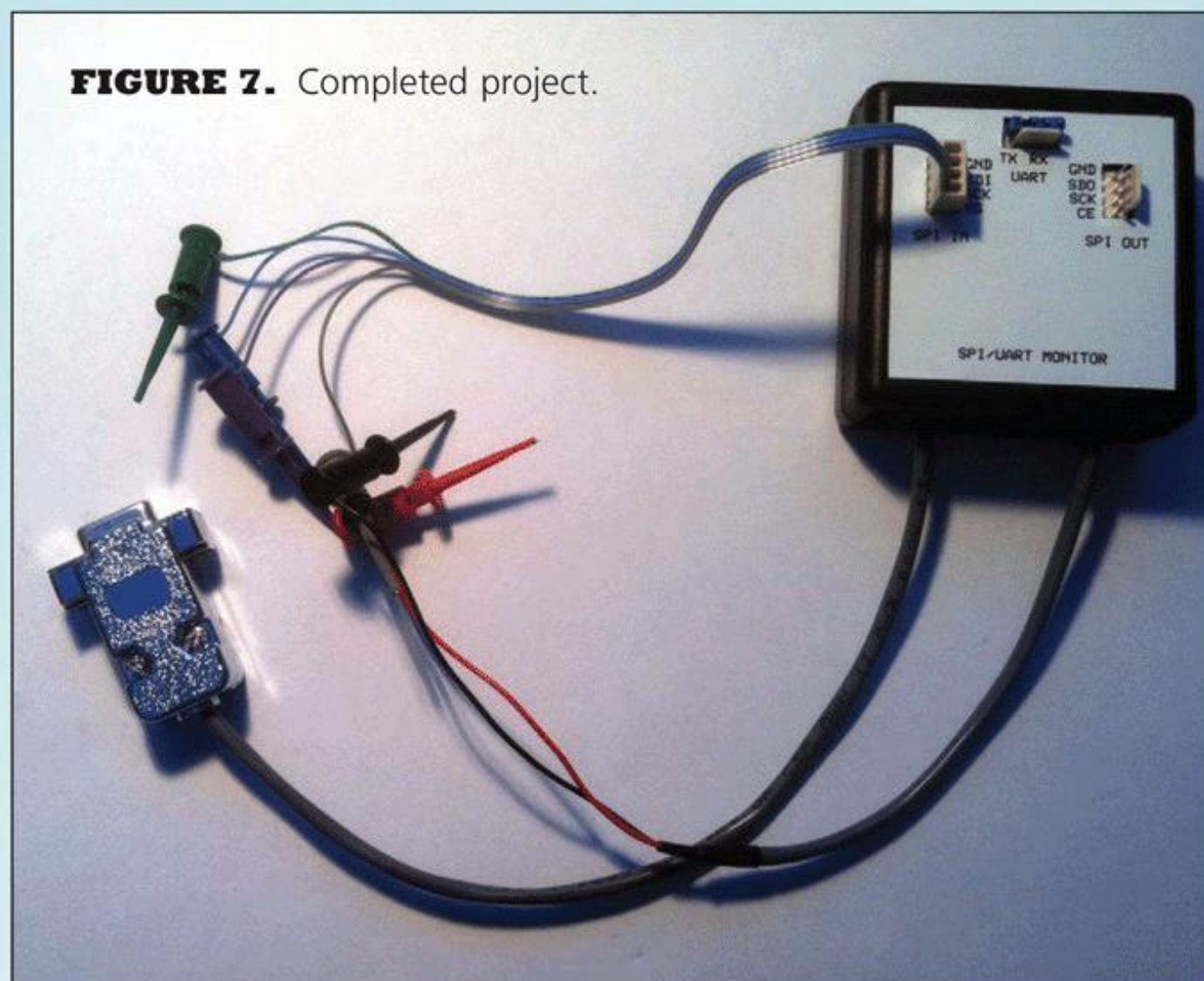


FIGURE 7. Completed project.

```
// -----
// SPI Interrupt handler
// -----
// this function manages incoming spi traffic and puts it in the fifo
// -----

#include <pic.h>

void SSP_isr() {
    // handle activity on sync serial port
    // if spi byte is ready...
    if (spi_data_is_in()) {
        // get the byte and stick it in the data buffer (FIFO)
        DataBuffer[InPos] = spi_read();
        // confirm that we have not exceeded the buffer size, if so wrap around to start
        if (InPos > MAXQUEUE SIZE)
            InPos = 0;
        else
            InPos++;
        // add this item to count (keeps track of bytes in queue)
        ItemCount++;
    }
}
```

FIGURE 8. FIFO fill code.

SPI Mode Configuration

The modes for the SPI bus can be a bit confusing partly because of the terminology, but it is really quite straightforward. There are four basic operating modes logically named: 0, 1, 2, and 3. These simply refer to the combinations of whether the clock signal idles low or idles high, and whether the data is latched into the receiving device on the rising edge or falling edge of the clock. (Please refer to **Figure A** for discussion.)

When data is sent out of the SPI port, the clock starts out in an idle state. This state is set by the CPOL signal. This can be seen in **Figure A**, where the CPOL signal is 0. The clock starts out low and ends low, and when the CPOL is 1, the clock starts out high and ends high.

When data is sent out of the SPI port, the clock edge selected by CPHA latches the data in the middle of the active data time using either the rising edge (CPHA = 0) or falling edge (CPHA = 1). In addition, this defines when the data line transitions to the next bit (the X part of the waveform in **Figure A**) which occurs on the opposite edge of where the data is latched on.

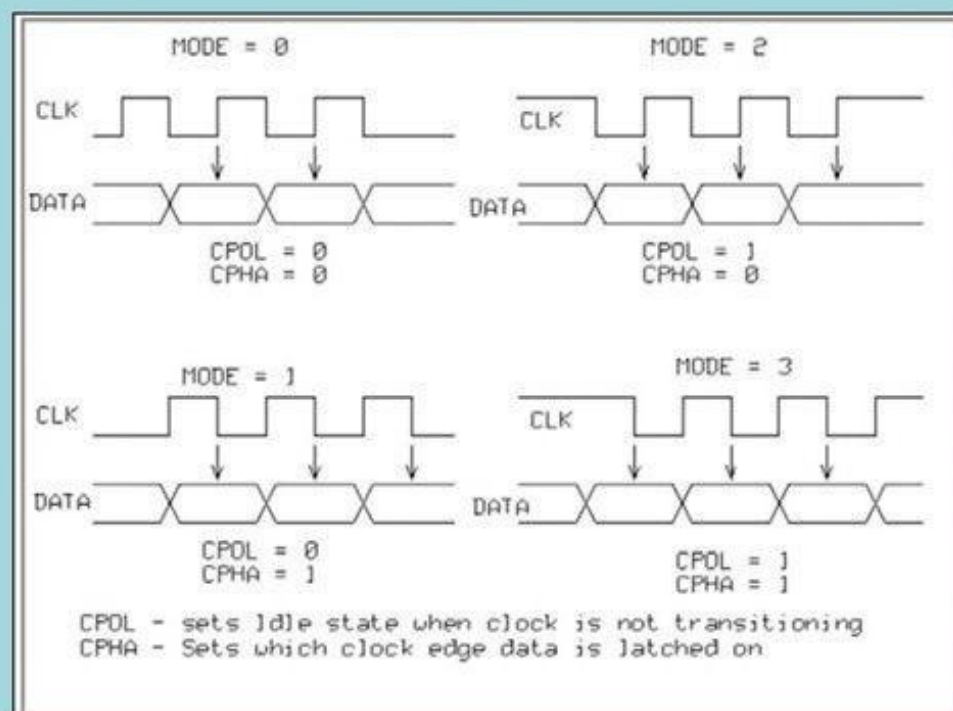


FIGURE A.

buffer is a perfect choice for this feature. The SPI data fills the FIFO by a simple read into the FIFO when the bytes appear at the SPI input port. The FIFO is drained by the program as it sends the data out to the PC via the PIC's UART using printf statements. I chose to put the SPI FIFO filling portion in an interrupt handler which is very easy to use in the CCS environment. The FIFO drain capability resides right in the main loop of code (in fact, it does little else). The code sections are shown in **Figure 8** and **Figure 9**.

The FIFO is implemented as a buffer of 60 bytes called DataBuffer. The incoming SPI byte is inserted into the DataBuffer (an array) at the current location pointed to by the InPos index. After this, the index pointer is checked to see if it is at the end of the FIFO and if so, the index is wrapped back to the top position in the FIFO (set to 0).

The drain operation works similarly, except the data is removed from the currently pointed to location via OutPos. OutPos is then checked to see if it is at the end and if so, it is reset to the top. This is elaborated on in **Figure 10**. That pretty much covers the FIFO part.

The next item to handle was the configuration of the SPI bus. It seems to me that there is always one element of a project that appears to be straightforward and then turns into a snake pit. That was the case here. I wanted to implement the SPI configuration as fully as I could, so the goal was to cover all of the four operating modes in the SPI bus standard. These modes are covered in more detail in the **sidebar** "SPI Modes."

Because I had implemented the SPI receiver on the hardware SPI port, the setup of CPOL and CPHA signalling was just a matter of setting the appropriate bits in the registers for that port. Microchip has kind of a funny way of labeling these signals in the registers which required significant noodling to get right (and Googling). On the transmitter side, I had implemented the port using the built-in functions for a soft SPI interface. I was not able to figure out how to change the settings of the port at run time since the port configuration is done through a compiler directive. So, in order to complete my mission, I had to write a bit-banged SPI transmitter which appears in

```
// loop forever
while(FOREVER)
{
    // output the current byte in the output count
    spiTx(OutByte);
    delay_ms(1);

    // inc output byte to next value
    OutByte++;

    // This code implements the drain function of the FIFO
    // check if a byte is available in the fifo (itemcount >0)

    if(ItemCount >0)
    {
        // be sure we have not exceed the buffer size, if so report
        if(ItemCount > MAXQUEUE SIZE)
        {
            // we have a buffer Overrun
            printf("Buffer Overrun\r\n");
        }
        // format output by inserting a new line if 16 bytes have been printed
        NewLineCount++;
        if(NewLineCount >= 16)
        {
            // output the byte out the uart plus a return to align to 16 symbols wide printout
            printf("%x\r\n", DataBuffer[OutPos]);
            NewLineCount = 0;
        }
        else
        {
            // output the byte out the uart
            printf("%x", DataBuffer[OutPos]);
        }
        // remove the last byte printed from th buffer (item count and reposition buffer pointers)
        ItemCount--;
        if(OutPos > MAXQUEUE SIZE)
            OutPos = 0;
        else
            OutPos++;
    }
}
```

FIGURE 9. FIFO drain code.

the listing in **Figure 11**.

After considerable head scratching, I managed to get the bits on and off at the right times and voila ... data was flowing. I tested the process by pumping out the data from the SPI port and looping it back to the receive side. I also monitored the data lines with a scope.

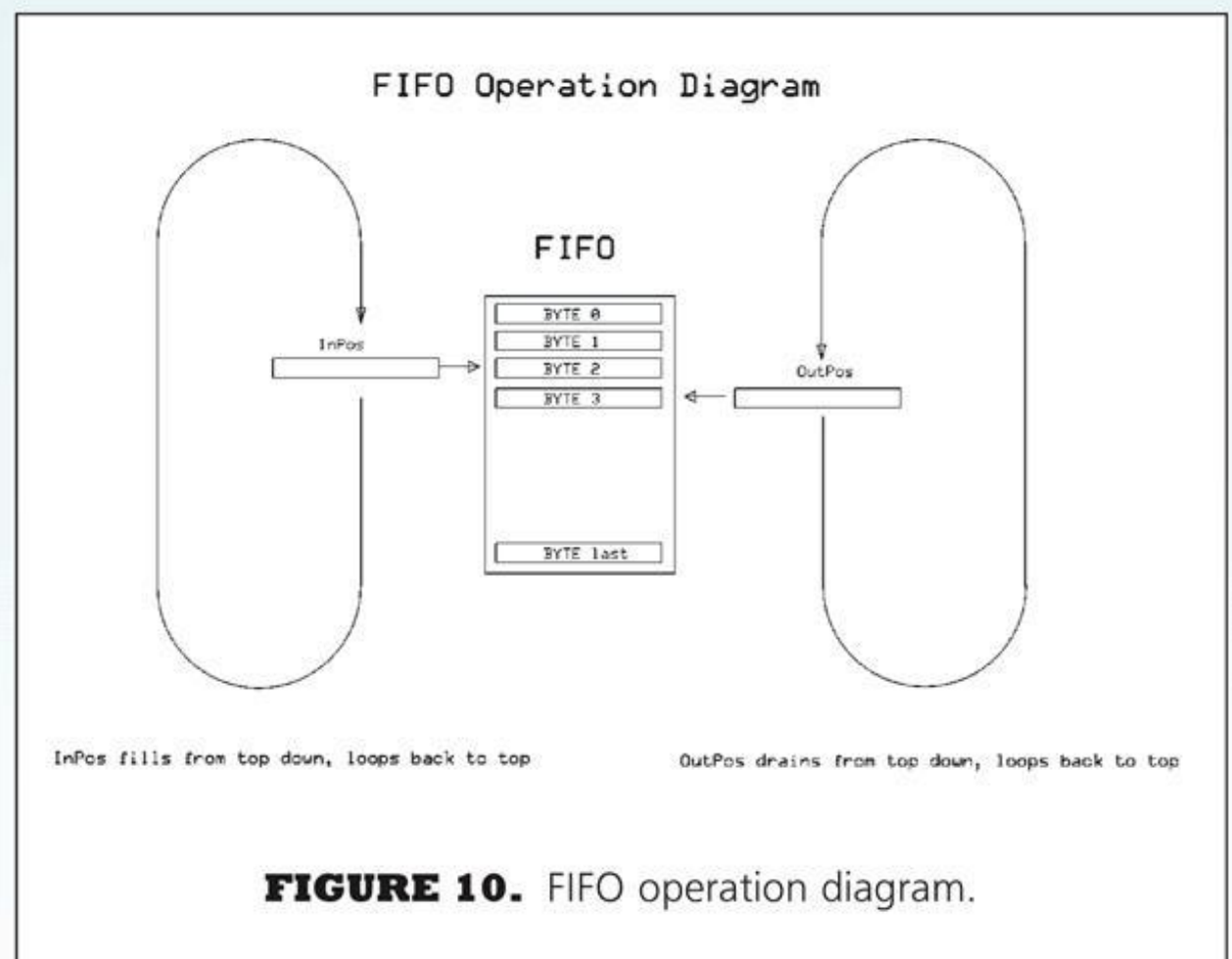


FIGURE 10. FIFO operation diagram.


```

// check clock polarity desired, this determines the starting clock level
// if cpol is 0 the clock idles low.
if(cpol == 0)
{
    output_bit(CLK,0);
}
else
{
    output_bit(CLK,1);
}
// init the bit counter
Bit = 0;
// set up the data (DOUT) pin to be the same as bit 0 of data
output_bit(DOUT,bit_test(data,Bit));
// drive the chip enable line low
output_bit(CE,0);
// delay some setup time
delay_us(100);

// Loop through each bit toggling the clock and updating the data bit such that the
// data is valid on the rising edge for cpha = 0
if( cpha == 0)
{
    while(Bit < 8)
    {
        output_toggle(CLK);
        delay_us(100);
        Bit++;
        output_bit(DOUT,bit_test(data, (Bit)));
        output_toggle(CLK);
        delay_us(100);
    }
}
else
{
    // Loop through each bit toggling the clock and updating the data bit such that the
    // data is valid on the falling edge for cpha = 1
    while(Bit < 8)
    {
        output_toggle(CLK);
        delay_us(100);
        output_toggle(CLK);
        delay_us(100);
        Bit++;
        output_bit(DOUT,bit_test(data,Bit));
    }
}
// when all 8 bits are done finish off
output_bit(DOUT,0);
output_bit(CE,1);
}

```

FIGURE 11.
Implementation of
bit-banged SPI port.

The last item was to configure the SPI settings from the terminal — perhaps the world’s tiniest and simplest user interface. I created a small set of commands to set the various operating modes and printed out a very short menu to the terminal. This is shown in **Figure 12**. Upon reset, the menu is displayed and the program waits until you enter a command, which either sets the mode of the SPI bus or initiates operation. The monitoring proceeds until the user resets the processor (through a power cycle).

Future Expansion

I²C functionality could be added to the system quite easily by tweaking the hardware for the SPI port to include appropriate pull-ups and updating the software to read the I²C protocol. The prototype boards allow easy rip-up and replace so this would not be a big deal.

The current processor only has 256 bytes of RAM, but other devices are available up to 1,024 bytes and an internal oscillator of 32 MHz. Moving into the PIC 18F family or perhaps an Arduino would allow for a much more powerful system.

So, that concludes the SPI monitor project. Full details are available on the website if you want to construct your own. There is plenty of opportunity for improvement if you have any ideas. **SV**

FIGURE 12.
User interface
menu.

```

Received/Sent data
SPI/UART Monitor Ver 2.0

Mode 0 Set --- Clk idle low, data transfer on rising edge
Commands -

m0<ret> - SPI Mode 1
m1<ret> - SPI Mode 1
m2<ret> - SPI Mode 1
m3<ret> - SPI Mode 1
gg<ret> - Go - start execution

gg
Cmd = g Code = g
Waiting for data...
|00|01|02|03|04|05|06|07|08|09|0a|0b|0c|0d|0e|0f
|10|11|12|13|14|15|16|17|18|19|1a|1b|1c|1d|1e|1f
|20|21|22|23|24|25|26|27|28|29|2a|2b|2c|2d|2e|2f
|30|31|32|33|34|35|36|37|38|39|3a|3b|3c|3d|3e|3f
|40|41|42|43|44|45|46|47|48|49|4a|4b|4c|4d|4e|4f

Modem lines
☒ CD ☒ RI ☒ DSR ☒ CTS ☐ DTR ☐ RTS

```

Sources

CCS Compiler
www.ccsinfo.com

Microchip
www.microchip.com

BoardworX System
www.boardworxsystem.com

Project Details
markmitchell@mx2systems.com

mymindsi.com

BUILD ANYTHING

Ignite your creativity and imagination with the all **new** MINDS-i robotics system. Finally, a rugged **quick-lock** construction system that empowers you to envision, build, and re-create anything that you can imagine with your own "mind's eye".



Patents US 7,517,270; US 7,410,225 B1; US 7,736,211; US 7,841,923; International Patents Pending.

DO ANYTHING

Utilize PCS Robotics controllers, sensors, and software to bring MINDS-i to life. Use PCS curriculum in the classroom to elevate your student's interest in science, technology, engineering and math!

imagine it



PRO-LITE

explore it



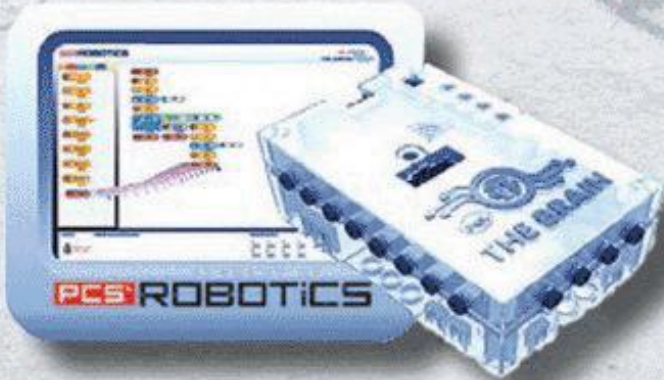
build it



play it



program it



pcsedu.com/servo

I²I Controls

by engineers, for engineers

All SPECTRUM ACE™ CPU boards contain ALEC™. With a built-in text editor, integrated debugging and trace, we make development easy.

SPECTRUM ACE™ and ALEC™ are the perfect combination of power and intellect. Program embedded applications with ease. Our expansion boards all have an ALEC™ command interface, so every environment is fair game.

Start your day with a hot cup of coffee and end it with a hot application!

www.i2icontrols.com

Whether you are a novice or an expert, this is a great way to get started!

The SPECTRUM LITE™ Evaluation Package comes with a SPECTRUM LITE™ CPU, the SPECTRUM LITE™ proto-board, and a 5V DC 1 amp power supply.



\$89.00

Building Bots From Found Parts

by Gordon McComb

“Found parts” are things you find around the house — or garage or hardware store or any place else — that are just *begging* to be used in your next robot. Found parts can help reduce the costs of building a bot. Plus, if the part can be used as-is without cutting, it makes building the robot easier. You don’t need as many tools for construction.

www.servomagazine.com/index.php?/magazine/article/june2011_McComb

Toys are among the most popular form of found parts. If you’ve been building automaton for any length of time, no doubt you’ve got a few repurposed K’NEX or Erector sets driving around your living room. There’s virtually no limit to the number and type of found items you can use in your robotics projects — either as the body of the robot or as a main part.

In this article, I’ll talk about making a small robot using a pre-shaped piece of metal commonly found at any home improvement store. It’s already in the shape and size for a bot, so there’s nothing to cut. You’ll drill a few extra holes, mount some motors and a caster, and you’re done. Construction time is 30 minutes or less.

Introducing the “No-cut” Robot Base Philosophy

Of all the aspects of robot building, cutting stuff up is my least favorite — especially if it involves metal. Many

robot designs use stock metal of some kind: U-channel, tubing, strips, or large plates that must be cut down to size.

But what if you could find metal already in the shape you need for building your robot? You can, but this stuff is found in a different part of the hardware store than the stock metal bins. With it, you can construct “no-cut” metal platforms that require no — or at the least, very little — cutting.

The basic idea behind the no-cut is to use base materials that are already at or very near the proper size and shape. The parts of the robot — the motors, sensors, batteries, and so forth — can then be attached using fasteners, glue, hook-and-loop, double-sided foam tape, tie-wraps, or other techniques. The concept of the no-cut seems simple (and obvious) enough, but it does require thinking out of the box to identify ready-made things that you can repurpose.

The Mini T-Bot we’ll describe here is an example of a no-cut mobile robot. It uses a readily available — and

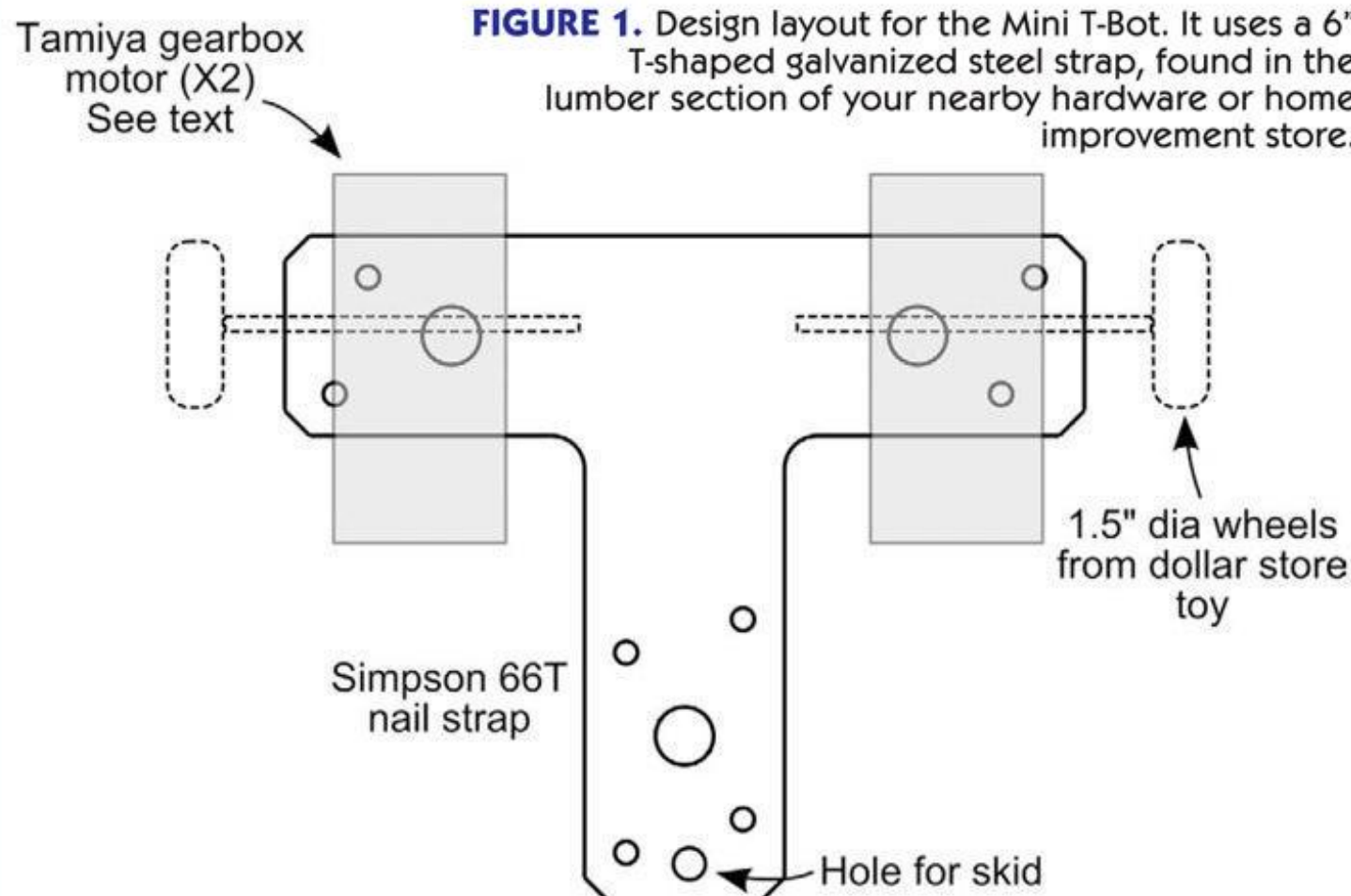
inexpensively priced — 6” strapping T (also spelled tee), commonly used to lash together pieces of lumber in a home. The basic layout for the Mini T-Bot is shown in **Figure 1**.

Look for strapping Ts in the lumber section of your local hardware or home improvement store. Strapping Ts are available in numerous sizes; the 6” size is the smallest that I’ve seen, but they are also available up to 16”. The size measures the crossbar at the top of the T; the vertical or stem portion of the T is in various lengths, depending on design.

One popular strapping T is the Simpson Strong-Tie T Strap; I purchased mine at the local Home Depot, but they are sold at many local and online hardware stores. The brand doesn’t matter; anything similar will do.

The Mini T-Bot uses the Simpson

FIGURE 1. Design layout for the Mini T-Bot. It uses a 6” T-shaped galvanized steel strap, found in the lumber section of your nearby hardware or home improvement store.



66T, made of 14 gauge galvanized steel. The T measures 6" x 5" with a strap width of 1-1/2". The 66T — like most strapping Ts — already has holes in it for nailing. The holes are offset and most will not line up with hardware you want to hang on your robot, so you'll need to drill new holes. A power drill (or better yet, a drill press) is recommended for drilling the holes.

Constructing the Mini T-Bot

The Mini T-Bot requires a few extra parts in addition to the strapping T and assorted fastening hardware. You are, of course, free to substitute these for others you may have on hand or like better.

You need two each of the following:

- Tamiya three-speed crank axle gearbox motor, #70093.
- Plastic wheels from a dollar store toy; 1-1/2" to 2" diameter, 2 mm bore hub.

The Tamiya parts can be purchased from most any online hobby retailer; see **Sources** for a selected list. The motors are mounted on the ends of the crossbar. To keep the cost of construction to a bare minimum, I've used a non-rotating skid for balancing the other end of the robot. The skid is nothing more than a 6-32 x 1/2" machine screw, hex nut, and steel or nylon acorn (cap) nut. The roundness of the acorn nut provides a smooth gliding surface.

For your reference, the finished Mini T-Bot is shown in **Figure 2**.

Begin by assembling the two gear motors according to their included instructions. With the Tamiya three-speed crank axle gearbox, you can select any of three gear ratios: 17:1, 58:1, and 204:1. The 204:1 ratio makes for a fairly slow bot, but it offers the most power for trekking through dense carpet. Try the 58:1 speed, which should be acceptable as long as the robot isn't unnecessarily weighed down. You can always disassemble the motors and change the gear ratio.

After building the gearboxes, don't snap in the motors just yet. Use the included tube of grease to lubricate the gears. Be sure to wash your hands to remove any grease residue before continuing with the rest of the construction.

For my prototype, I used a pair of 1-1/2" diameter (1/4" tread width) wheels I pulled off a toy purchased at the nearby dollar store. The bore in the hub of the wheel was already just the right diameter for the 3 mm hex axle used with the gearbox motors. If your wheels have a small bore, carefully drill it out to the proper size. Don't make it too large, or else the axle will simply spin inside the wheel.

Cut the axle to about 2-3/8" in length. Use a small hammer to gently tap the axle into the bore of the wheel. Then, secure the axle inside the motor using the included setscrew. Use your fingers to manually rotate the gears so that the setscrew is accessible from the top of the motor

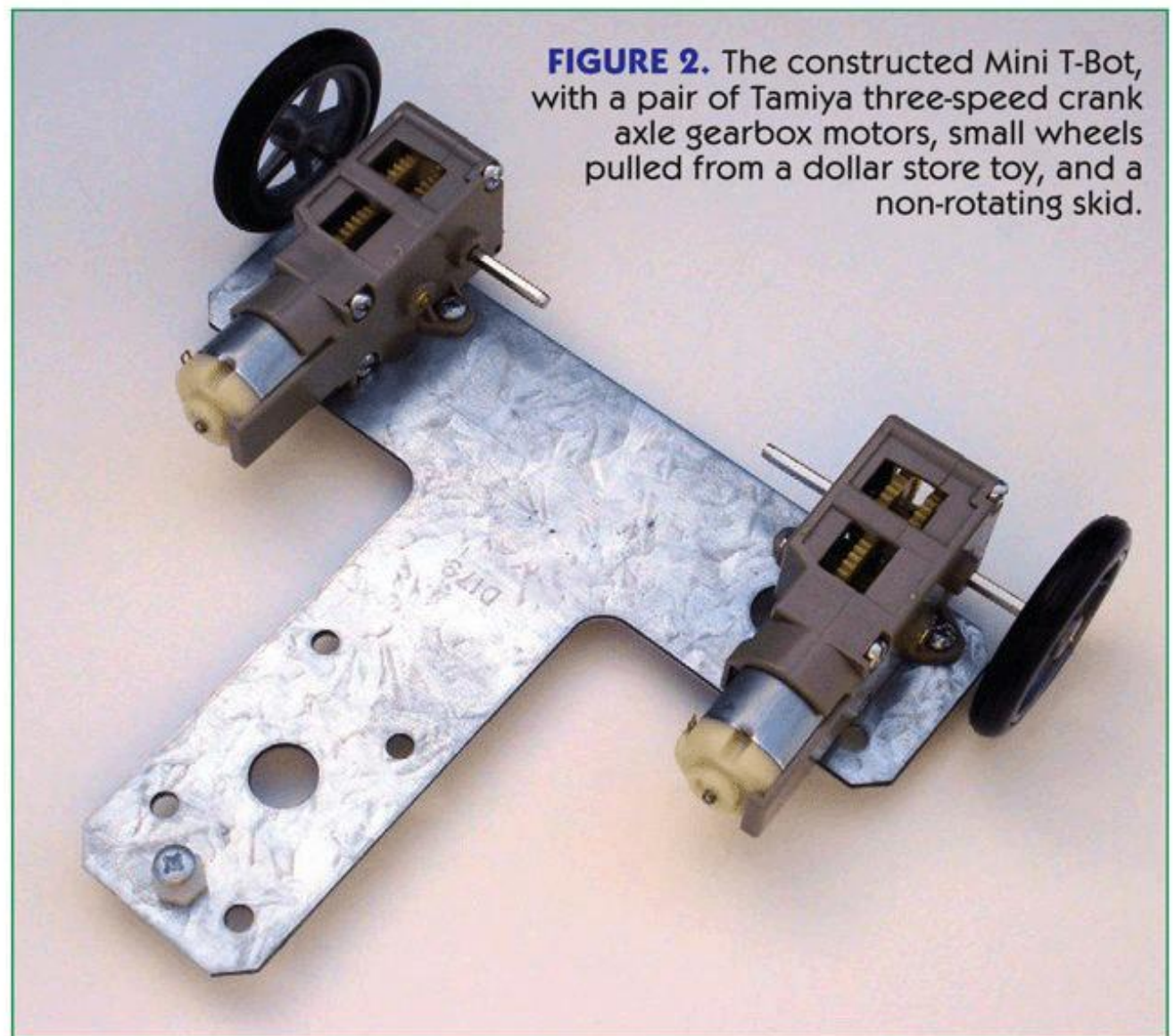


FIGURE 2. The constructed Mini T-Bot, with a pair of Tamiya three-speed crank axle gearbox motors, small wheels pulled from a dollar store toy, and a non-rotating skid.

(the bottom has the two flange eyelets for mounting).

Slide the axle so that there's about 5/8" clearance between the side of the motor and the inside rim of the wheel (see **Figure 3**). Use the included hex key wrench to tighten (but don't over-tighten!) the setscrew to hold the axle in place. Remember that you're constructing a left and right motor. The difference is whether the wheel is attached to the left or right side of the axle.

Here are selected sources for the Tamiya motors, wheels, and ball caster detailed for the Mini T-Bot, plus online sellers of motor bridge modules:

Parallax - www.parallax.com
Motor bridge modules.

Pololu - www.pololu.com

Tamiya motors, wheels, and other accessories; selection of motor bridge modules; high efficiency replacements for FA-130 motors.

Robotshop - www.robotshop.com

Tamiya motors, wheels, and other accessories; DFRobot Arduino motor bridge; other motor bridge modules.

Robot Store/Jameco - www.robotstore.com

Selection of Tamiya motors and wheels; L298 motor bridge chip; and assorted components.

Solarbotics - www.solarbotics.com

Tamiya motors, wheels, and accessories; motor bridge modules; high efficiency replacements for FA-130 motors.

SparkFun - www.sparkfun.com

Motor bridge modules; breakout boards; and chip-level controllers.

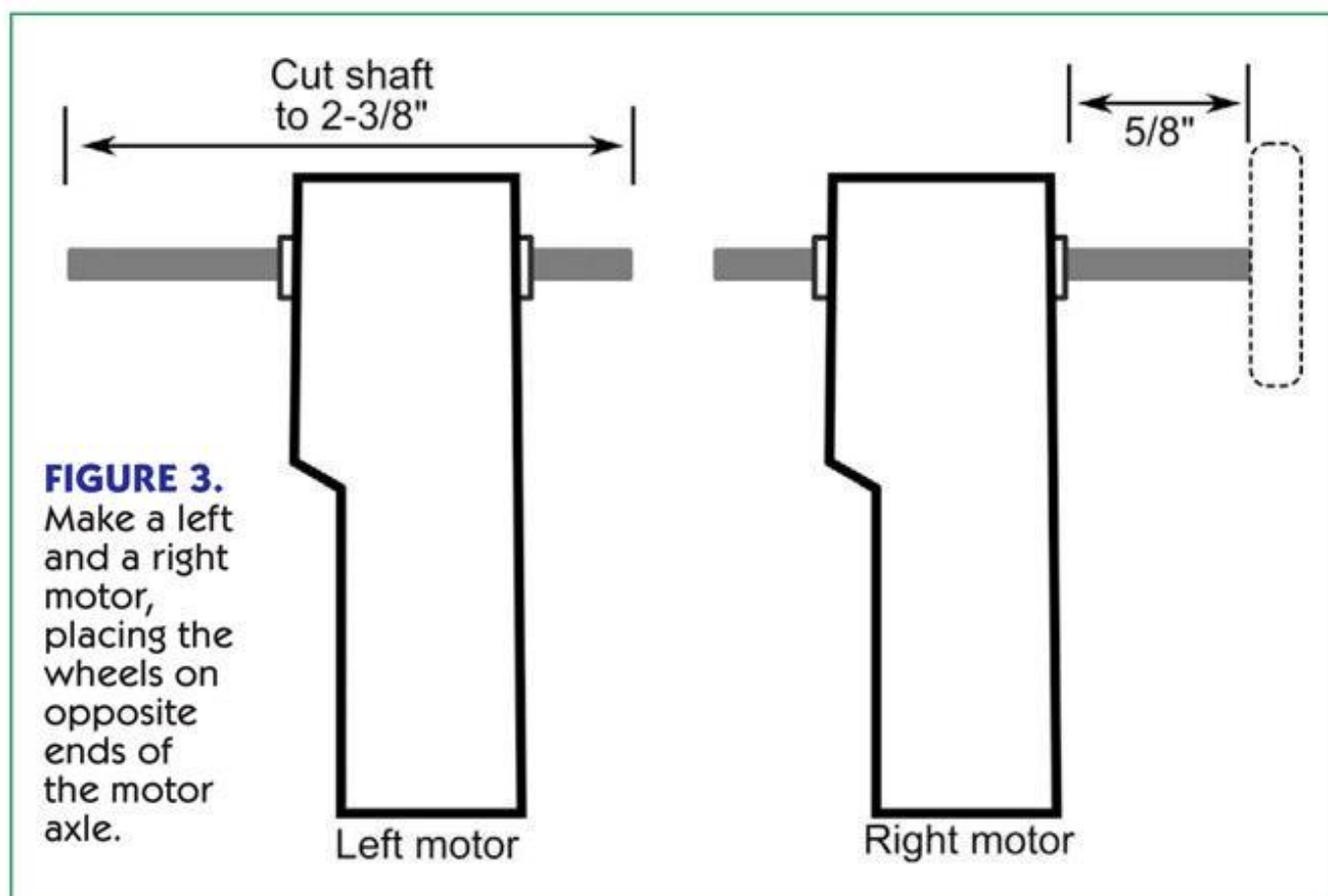
The Robot Marketplace - www.robotmarketplace.com

Tamiya motors, wheels, and other accessories; motor controllers.

Tower Hobbies - www.towerhobbies.com

Full line of Tamiya products.

Sources



Only a few holes need to be drilled in the strapping T to secure the motors. Use a 9/64" drill bit to make holes for 4-40 x 1/4" or 4-40 x 3/8" machine screws. The small fasteners and the somewhat larger holes provide some "slop" in mounting.

For each motor, position it near the end of the crossbar where you want it to go. Use the mounting flanges of the motors to mark the holes for drilling. There's already some holes drilled into the T, and you may be able to use one or two as a starting point; you'll need to drill the others to match the motor mounting flange.

After marking, use a center punch (the spring loaded kind is easiest) to indent a spot for drilling. The small dimple created by the punch helps prevent the drill bit from "skating" across the metal.

For best results, use a drill press and clamp the T in a vise. (Avoid holding the T with your hands because if it's yanked loose during drilling, it could cause an injury).

Place a block of soft wood (such as pine) behind the metal for support, and be sure to wear eye protection. Set the drill to slow speed to avoid overheating the bit. Add a drop of oil over the punched mark, and slowly drill through the metal.

Drill a single hole — also using a 9/64" bit — near the bottom of the T stem for the skid. Do the same for this hole as above: mark with a center punch, add a drop of oil,

and drill slowly. Wipe off the oil when you're done drilling.

Secure the motors using four 4-40 machine screws and nuts. The heads of the screws should be on the side of the motors; the nuts should be on the underside of the robot. Because the holes are somewhat oversized, you can use the slop to help align the motors. You want to make sure the wheels are as parallel as possible.

Finally, construct the skid by threading a 6-32 hex nut onto a 6-32 x 1/2" machine screw. See **Figure 4** for the assembly detail. Place the hex nut near the head of the screw. Insert the screw into the hole and fasten in place with a 6-32 acorn nut on the bottom side of the base. Tighten the hex nut against the T to secure the screw in place.

Upgrading the Gearbox Motors

The DC motors used in the Tamiya three-speed crank axle gearbox are designed for three volt operation, and they are relatively inefficient. They're fine if you plan on controlling your Mini T-Bot using switches or relays and a three volt battery pack, but they're not a good choice for electronic motor control. Powering them at the usual 4.5 to 6 volts will shorten their life, plus require motor bridge circuitry capable of supporting three to four amps (yes, amps!) of current if the motor becomes stalled (it stops and won't move).

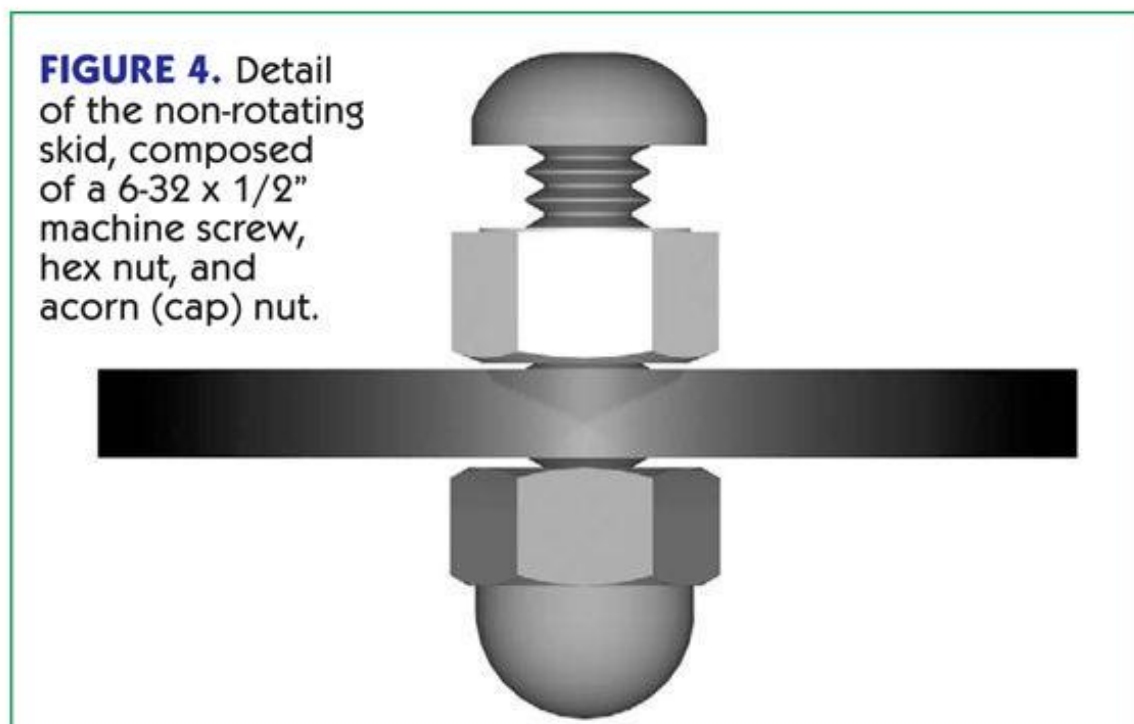
The three-speed crank axle gearbox uses a Mabuchi FA-130 motor. The 130 size is common (it's popular in slot cars), and you can get replacement motors that are not only rated at higher voltages, but consume a lot less current. One such option is the 130-size motor offered by Pololu (their item #1117). It's rated at six volts, and has a stall current of only 800 mA. The price is under \$2 per motor.

The lower current motors not only consume less battery power, but allow the motor to be used with low cost H-bridge control circuitry. Suitable single chip solutions include the SN754410, though I prefer using a L298 as it provides a bit more functionality. (More about interfacing the Mini T-Bot to an L298 below.)

Enhancing the Mini T-Bot

There's no requirement that you design your Mini T-Bot with the same Tamiya three-speed crank axle gearbox motors and dollar store toy wheels. As an example, **Figure 5** shows an alternative design, using a collection of readily available Tamiya parts:

- *Tamiya #72004 worm gearmotors.* These provide gear ratios of either 216:1 or 336:1, making the robot slow but strong. I usually select the 216:1 ratio when constructing small desktop bots. You need two motors. (The mounting flange spacing is the same as



on the three-speed crank axle gearbox.)

- *Tamiya #70145 narrow tire set.* These are 58 millimeters (about 2-1/4") in diameter, and securely attach to the round 4 mm axle shaft of the worm gearmotors. The wheels come in pairs, so you need just one set.
- *Tamiya #70144 ball caster.* Use this instead of the static skid to provide a unidirectional caster. You can select any of four heights; when used with the above motors and wheels, pick the 16 mm height option. Drill either two or three holes for mounting the caster; the placement of the holes is somewhat critical as there is little room for error. You get two ball casters in the set, but you only need one.

The worm gears use Mabuchi RE-260 motors which (when operated at six volts) consume about 3 amps of current if stalled. That's too much even for an L298, so if you go this route you'll want to consider using more robust H-bridge circuitry.

Programming the Mini T-Bot

With your Mini T-Bot constructed, you'll want to watch it scoot around the floor. Your options are virtually

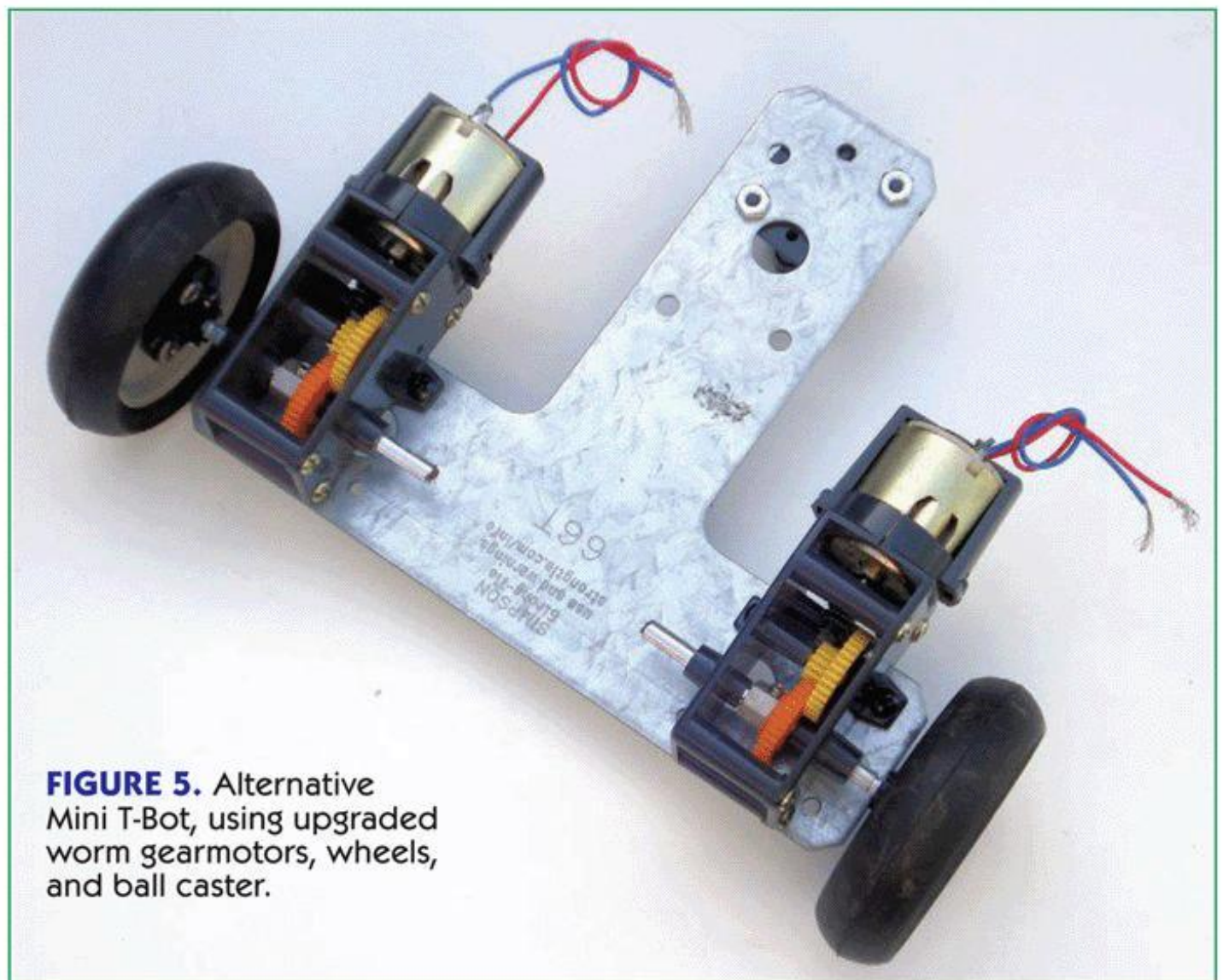


FIGURE 5. Alternative Mini T-Bot, using upgraded worm gearmotors, wheels, and ball caster.

unlimited, but for demonstration purposes, I'll discuss some simple Arduino code you can use to run your Mini T-Bot through its paces.

Remember that the standard Arduino cannot be directly connected to DC motors. You need to interface the motors to the Arduino through an H-bridge circuit, shield, or chip. **Figure 6** shows a typical H-bridge module that supports the direction and control of two DC motors.

For my prototype, I used a motor shield equipped with

A Menagerie of Found Parts

There are plenty of everyday objects you can use for robot building — all it takes is looking at them a bit differently. Here are some examples to whet your appetite:

Plastic storage containers. Available in square, round, and other shapes, these durable plastic boxes — available in the housewares section of any department store — can be used with or without their press-on lid. Containers are available from small snack size to big shoeboxes.

Plastic fishing tackle box. Wheels are mounted to the sides of the box; motors, batteries, and other critical components are placed inside. The box can be popped open to gain internal access.

Small "dorm-size" trashcans. Just large enough to hold a Big Gulp, these trashcans have a convenient cylindrical shape and removable top. Great for building miniature R2-D2 bots.

Computer mice. A discarded computer mouse makes a great body for a micro-miniature robot. Almost all mice can be disassembled by removing one or two screws on the bottom. Take out the innards and install small motors, a small battery, and a one-chip brain.

Compact discs and DVDs. Save the world's landfills and use these 4.7" diameter discs for robot bases. Use care when drilling holes in the plastic; the material can shatter into very sharp pieces. If you need added strength, sandwich two discs together.

Solderless breadboards. Solderless breadboards are used to experiment with circuits before using more permanent solder and wire-wrap construction. Mount motors and wheels on the underside of your solderless breadboard, and you create a versatile and ever-changeable mobile robot.

Plastic project boxes. These boxes — sold by RadioShack and other electronics stores — are made to hold custom electronics projects. The boxes come with removable metal or plastic lids to allow access to the inside. The plastic is easily drilled for mounting stuff.

Clear or colored display domes. Also called a hemisphere or half-round dome, display domes can be purchased in sizes from about 2" to over 12" in diameter. The dome can be used as the body of the robot, or as a cover to protect its electronics. A "robotic ball" can be made by gluing two domes together.

PVC irrigation pipe. All forms of polygonal frames can be constructed using PVC irrigation pipe. Most hardware and plumber supply stores carry PVC pipe in various sizes and wall thicknesses.

Hardwood laminated flooring samples. Already in about the right size and shape for a small bot, these samples are made by laminating a thin veneer over a sheet of high density board. Thickness is about 1/4". Cut off the tongue-and-groove edges used to assemble the wood to make flooring. Round off the corners to keep the wood from chipping.

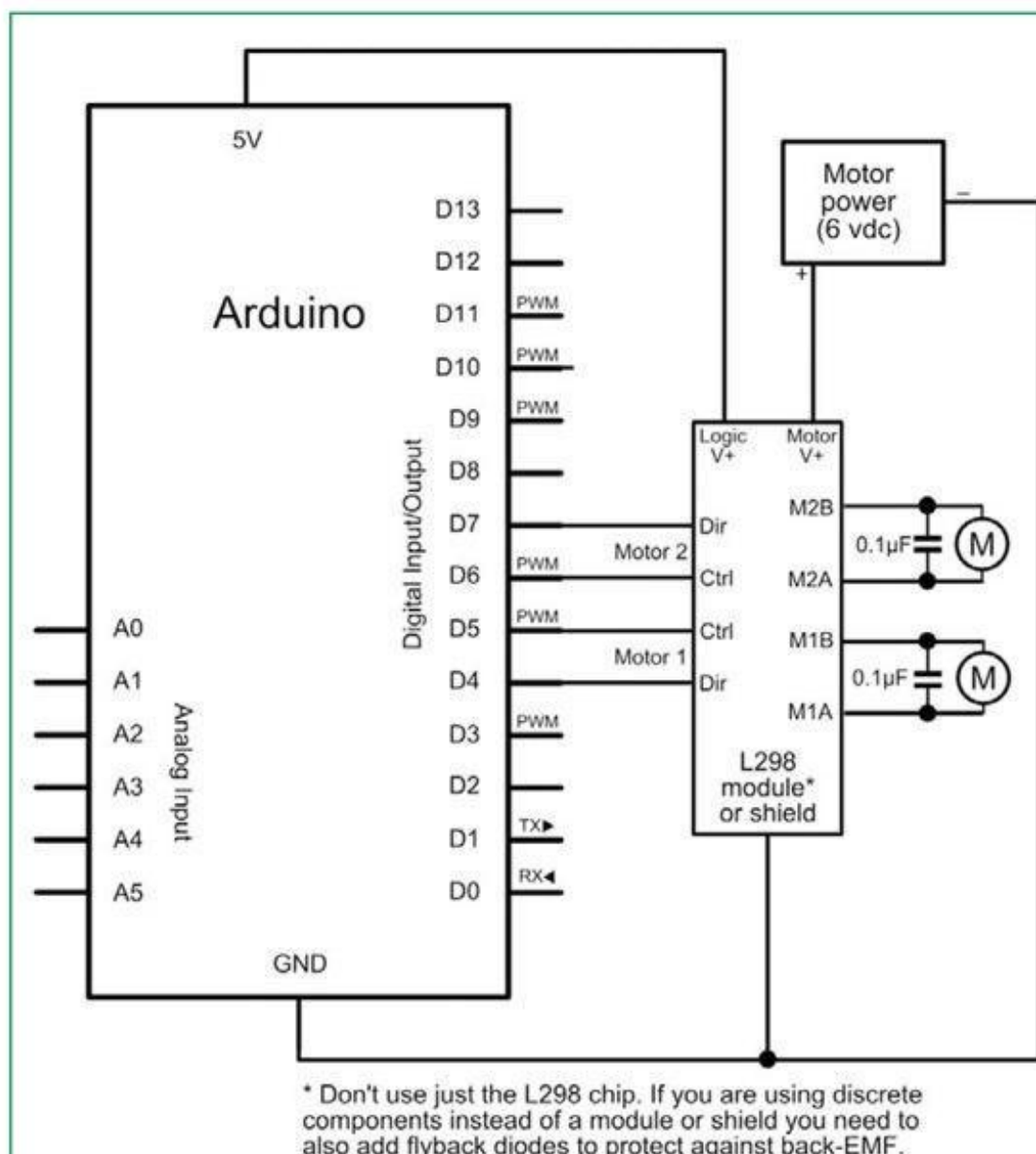


FIGURE 6. Basic wiring diagram for connecting an Arduino microcontroller to an H-bridge module. When using a shield, the electrical contact between the Arduino and module is incorporated in the edge connectors.

```

/*
  DC motor driver test
  Motors connected to digital pins
  4&5, and 6&7
*/

const int M1_Ctrl = 6;           //Set pin
                                  // assignments
const int M1_Dir = 7;
const int M2_Ctrl = 5;
const int M2_Dir = 4;

void setup() {
  pinMode(M1_Ctrl, OUTPUT);      //All pins are
                                  // OUTPUTS
  pinMode(M1_Dir, OUTPUT);
  pinMode(M2_Ctrl, OUTPUT);
  pinMode(M2_Dir, OUTPUT);
}

void loop() {                    //Loop through motion
                                  // routines

  robot_fwd();
  delay(3000);                   //Delay 3 seconds before
                                  // moving on

  robot_rev();
  delay(3000);

  robot_right();
  delay(3000);

  robot_left();
  delay(3000);

  robot_stop();
  delay(1500);
}

```

an L298 motor bridge IC. The shield I used is from DFRobot (available at **Robotstore.com**, among other online retailers). Another option is the DFRobot Romeo which combines an Arduino and the L298 shield in one unit.

The shield includes the L298 chip itself, plus the necessary flyback diodes to protect the L298 from back-EMF produced by the motors. Power for the motors is separate from the power to the Arduino and the logic circuitry on the shield.

(Note the addition of the 0.1 μF ceramic disc capacitors

at the motors. For best results, solder these directly to the terminals on the motor. The capacitors help to suppress noise caused by the motors which can interfere with the proper functioning of the microcontroller.)

There are a number of similar L298 Arduino shields and stand-alone modules, and if you use one of these, just remember to adjust the I/O connections accordingly. On the DFRobot shield, the motors are connected to pins 4/5 and 6/7. Each motor uses two pins: one for direction and one for control.

Listing 1 is a demo Arduino sketch that cycles through the basic functions of DC motor control for a robot: moving it forward and backward; steering it left or right; and stopping.

The control pin for each motor can also be used with pulse width modulation (PWM) to set the speed of the motor. **Listing 2** shows how this is done using the Arduino's *analogWrite* statement. To set the speed of the motor, you specify a number from 0 (off) to 255 (full on). These values control the duty cycle — the ratio between on and off times — of a fast series of pulses sent to the motor.

Not all motors respond to very low duty cycles. When used with the Tamiya worm gearmotors, values under about 96 have little or no effect. That's why the *for* loop in **Listing 2** starts at 96, and goes to 255.

Important note! In programming code, the physical direction of a DC motor is ambiguous. It all depends on how the motor is wired to the control circuitry. I intentionally wired both motors in the exact same way to the L298 shield which then required that the motors be commanded in opposite — one motor is set to LOW, while the other is set to HIGH. This propels the bot forward or backward. If a motor doesn't turn in the direction you want it to, merely flip its wiring to the control circuit.

Using Larger Ts for Larger Bots

You're not limited to just the 6" T strap. By using a bigger T, you can construct a larger (and heftier) robot. For your reference, **Table 1** are the specifications of the most commonly available sizes of Simpson Strong-Tie strapping Ts and their weight in ounces.

Yikes! ... the 1212T strap weighs almost a pound, so you'll need bigger motors (and batteries) to haul that kind of weight around.

The robot brute in **Figure 7** uses a pair of 12" straps, separated by 5" long aluminum tubing used as risers. In this

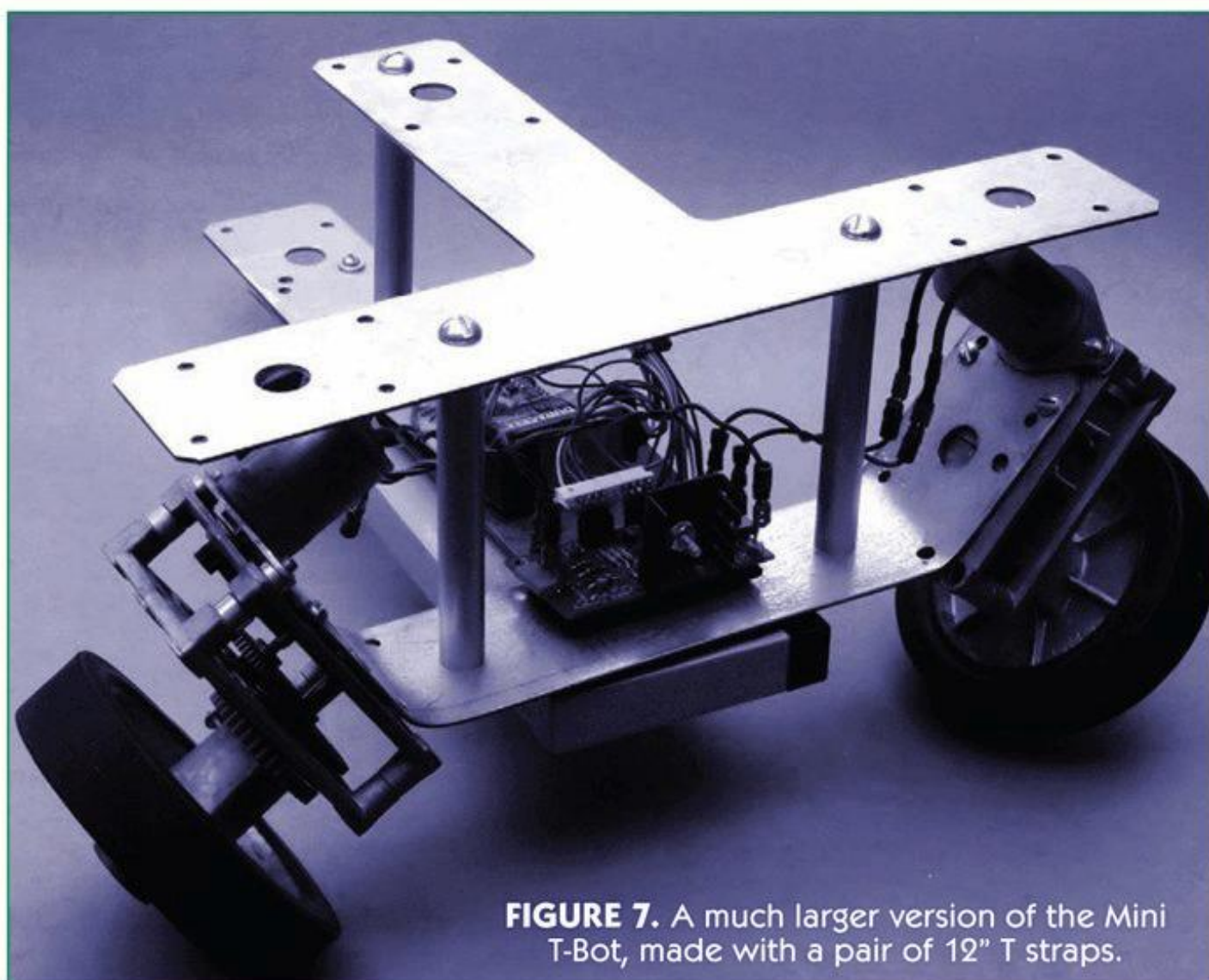


FIGURE 7. A much larger version of the Mini T-Bot, made with a pair of 12" T straps.

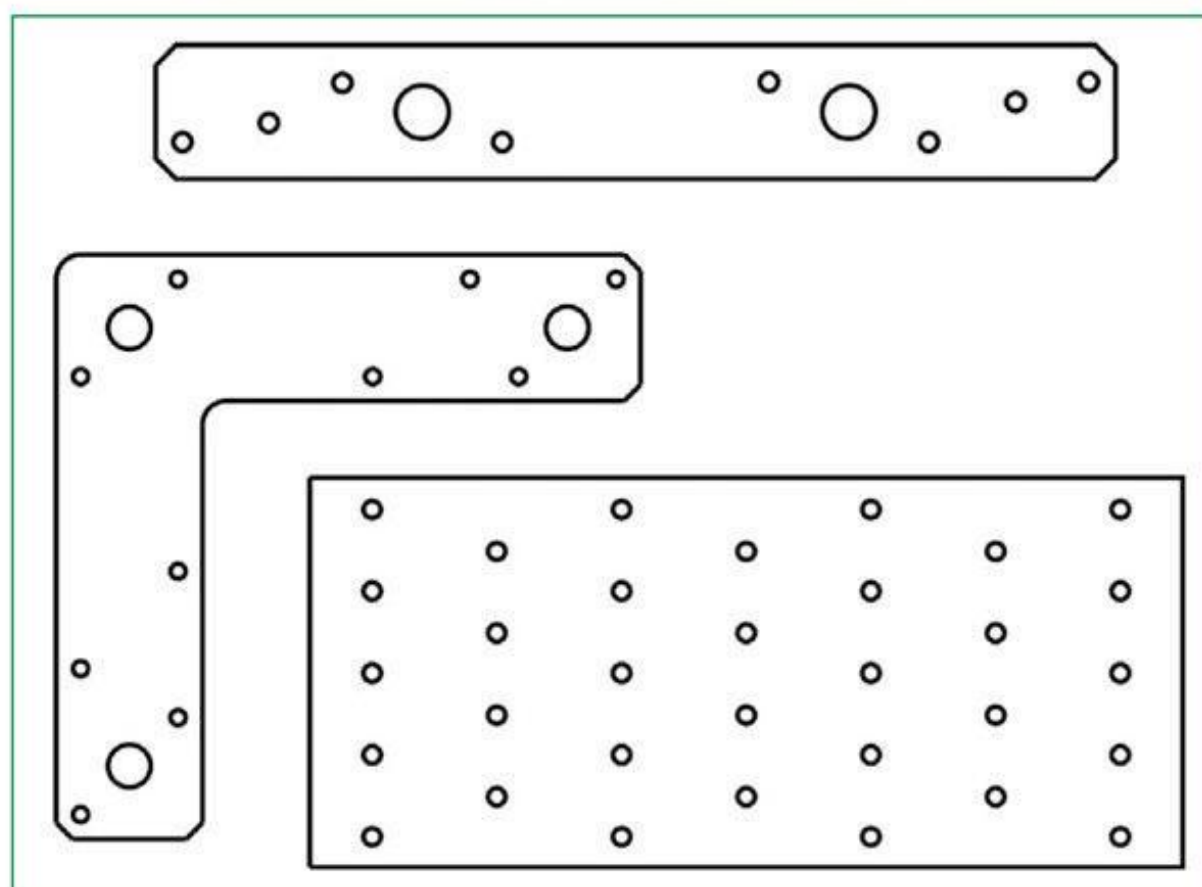


FIGURE 8. Outline shapes of three common nail plates. Like T straps, these are found in the lumber section of local hardware or home improvement stores.

particular prototype, the motors were mounted at an angle, with the metal of the lower T bent at 45°. This was partly done to accommodate the motor itself, as its mounting holes were on the side opposite the axle and wheels. I also did it simply to be different.

Other Sheet Metal for Lumber Strapping

Don't stop with just T-shaped straps for building robot bodies. The same lumber section of your local home improvement store has plenty of other choices. Some of it is specially formed and bent for things like hanging 2 x 4 joists in an attic or garage. These are somewhat less useful than flat metal.

Figure 8 shows the outline drawing of three commonly

Model	Material	L	H	W*	Weight
66T	14 gauge galvanized	6"	5"	1-1/2"	5 oz
128T	14 gauge galvanized	12"	8"	2"	11 oz
1212T	14 gauge galvanized	12"	12"	2"	14 oz

TABLE 1.

*Width is the width of the strapping metal.

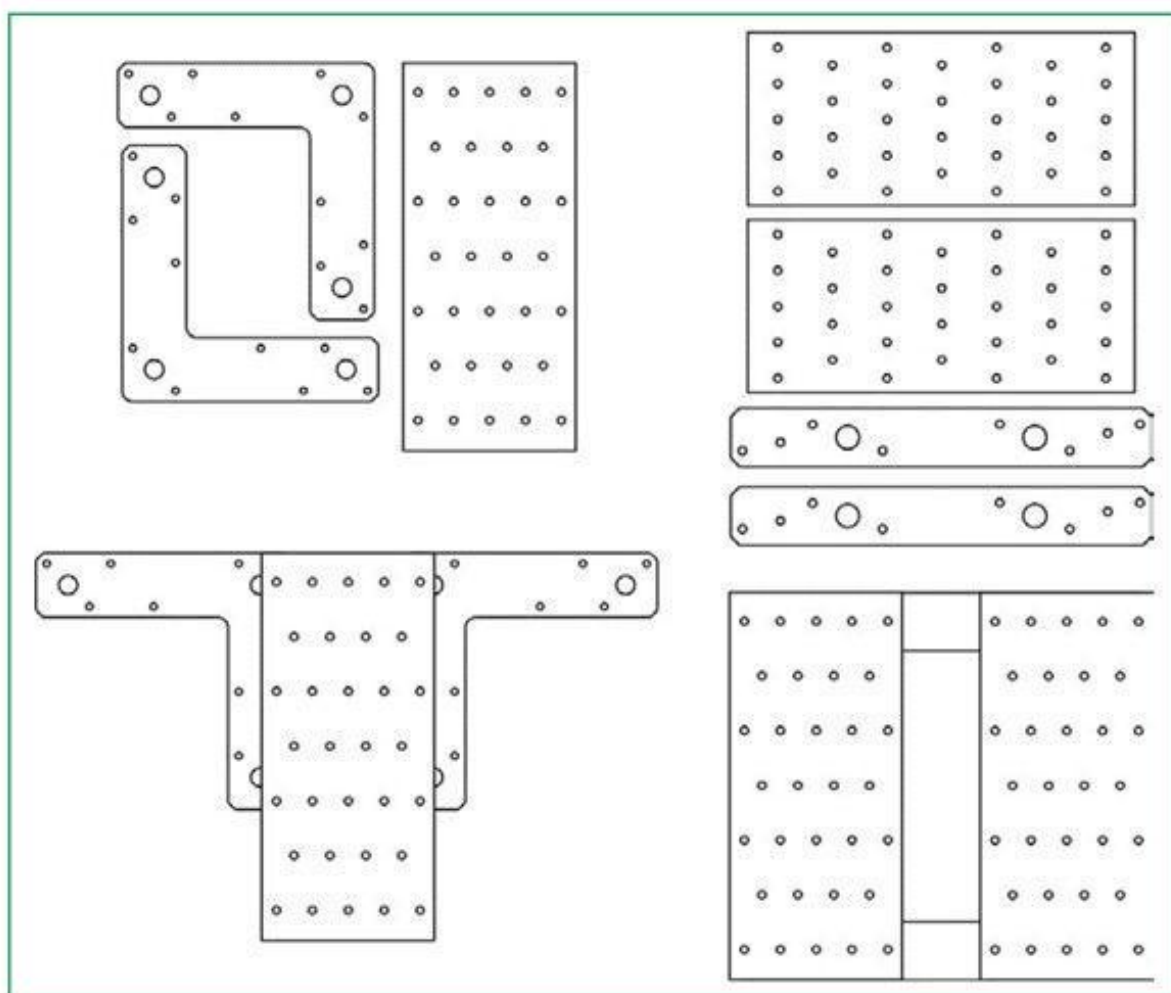


FIGURE 9. Two of an almost unlimited number of ways to combine nail plates to construct robot bases of all shapes and sizes.

available nail plates — so called because they're used to nail pieces of wood together. As with the T strap for the Mini T-Bot, these are made by Simpson. If you can't find this brand, there are other similar products out there.

LSTA9 Strap Tie measures 9" x 1-1/4". Example uses include: a center rail in a walking robot; a connecting strap

for wood, metal, or plastic bases; or a side angle bracket for tracked bases. 66L L Strap is an L-shaped plate that measures 6" on each side. If you need bigger, there's the 88L which is 8" on each side. Example uses include: mounting brackets for larger robots or outriggers for motors.

TP37 and similar Tie Plate is basically a flat plate with different lengths to suit various applications. The width for all sizes is 3-1/8". For the T35 length it's 5"; for the TP37 length it's 7", for the TP39 length it's 9". Example uses are: a robot base; a mounting plate for heavy parts (large motors, batteries) on framed robots; or side panels.

Figure 9 shows how you might combine some of these sheet metal pieces to make various kinds of robotic bases. Or, you could use the LSTA9 as a cross piece for mounting motors or legs, or use it as a long side bracket for a tank style robot.

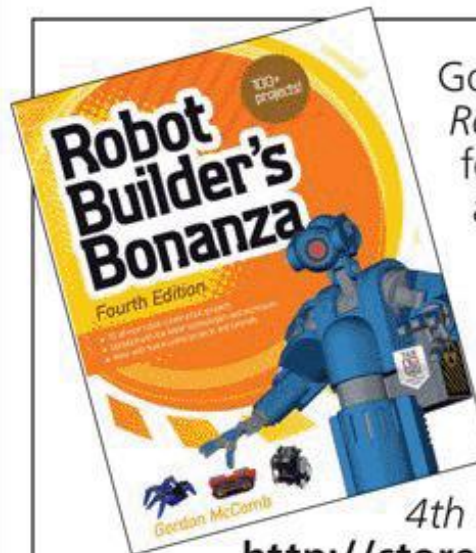
You may elect to cut or trim some of the pieces, but since they're already in the basic shape you need, there is less work required overall. Sheet metal for lumber strapping is typically 18 or 20 gauge, and can be cut with a hacksaw, metal snips, or — my preference — electrical or air powered motorized shears.

Of course, the concept of the no-cut extends beyond the Mini T-Bot or the other strapping products detailed here. You can use the same idea for other robot designs made with different metal materials, no matter where you find them. The key points to keep in mind are:

- The material should already be in the size you need, so no cutting is required.
- Avoid very thick materials for small robots, as they add unnecessary weight.
- Consider sheet materials that can be bent to create unusual robot base shapes. Try a 6" x 12" aluminum sheet purchased at a local hobby shop.

Keep Your Eyes Peeled and Your Tape Measure Out

Before leaving the home improvement store, be sure to take one last stroll down the aisles. You'd be surprised what you'll find when you look at things from a robot builder's perspective. You never know what interesting tidbits you'll discover that can be the framework of your next robot. **SV**



Gordon McComb is the author of *Robot Builder's Bonanza*, now in its fourth edition. Greatly expanded and updated, this best selling book covers the latest trends in amateur robotics, and comes with 10 all new robot construction projects, plus more ideas for building robots from found parts. Look for *Robot Builder's Bonanza*,

4th Ed in the SERVO Webstore at <http://store.servomagazine.com>. Gordon may be reached at rbb@robotoid.

Learn Electronics



Build

100's of Kits

Do Robotics



GSSTechEd.com
1-800-422-1100

CPLDs — Part 4

complex programmable logic devices

HDL Programming

by David A. Ward

As we've mentioned in the previous articles in this series, HDL (or hardware description language) is really the preferred method used to program CPLDs versus the graphical or schematic method that has been demonstrated so far. (We've used the schematic method up to this point because it's an easier way to get started programming CPLDs. Now that you have a better idea of what a CPLD is and what it can do, it's time to introduce and begin using HDL.

www.servomagazine.com/index.php?/magazine/article/june2011_Ward

Of course, in a single magazine article we'll only be able to give a fairly brief introduction to HDL. There are many resources available online if you want to learn more about HDL. I recommend *The Low-Carb VHDL Tutorial* by Bryan Mealy. This is a good and concise resource with many examples. HDL is also referred to as VHDL — the "V" stands for a very high speed integrated circuit. Although HDL may appear similar to some computer programming languages, the end results are quite different. When a computer language program is compiled, it is compiled into machine instructions that the microprocessor will execute one at a time in a sequential manner. When an HDL program is compiled, it is compiled into interconnections which will be configured into the CPLD. When the CPLD is programmed with these interconnections, the circuit will operate in a parallel or concurrent manner. So, the order in which HDL lines are entered into the HDL source code may not matter as far as the final operation of the circuit is concerned. HDL is really a description of a digital logic circuit rather than a set of sequential operations to be performed, such as in a computer program.

HDL programs contain two main sections: the "entity" section and the "architecture" section. The entity section comes before the architecture section and declares the inputs and outputs of the digital logic circuit. The architecture section describes how those inputs and outputs from the entity section will behave in a digital logic manner. **Figure 1** is an example HDL listing for a two input AND gate with inputs named A and B, and one output named X. As you examine the HDL listing, note a few things. First, HDL is not case sensitive, so that input "b" would be the same as input "B." HDL lines are terminated with a semicolon (;). White space — extra spaces added for clarity — are also ignored by the compiler. You can add comments where you want after placing two dashes (--); this means that the

compiler does not attempt to compile anything after the two dashes.

Let's take this simple HDL two input AND gate, compile it, and program it into a CPLD in the Xilinx Project

```
entity firstHDL is
    Port ( A : in  STD_LOGIC;
          B : in  STD_LOGIC;
          X : out  STD_LOGIC);
end firstHDL;

architecture Behavioral of firstHDL is
begin
    X <= A AND B;
end Behavioral;
```

FIGURE 1.

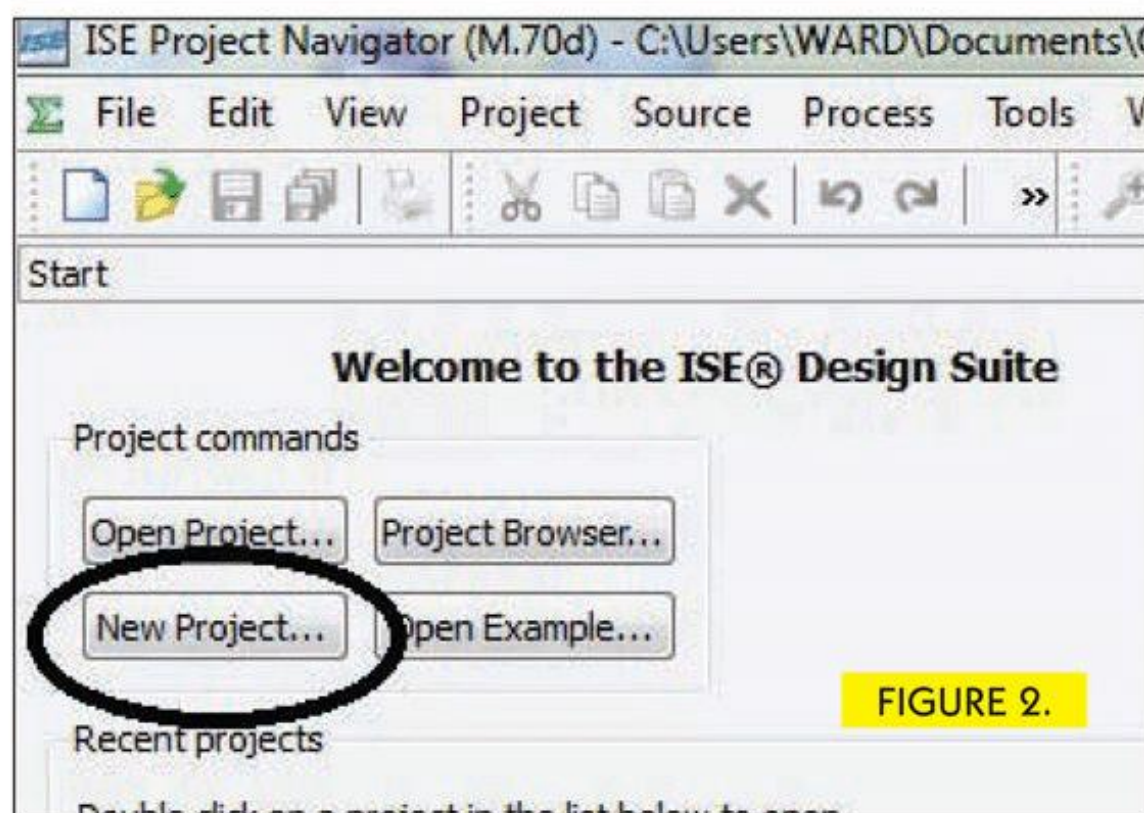


FIGURE 2.

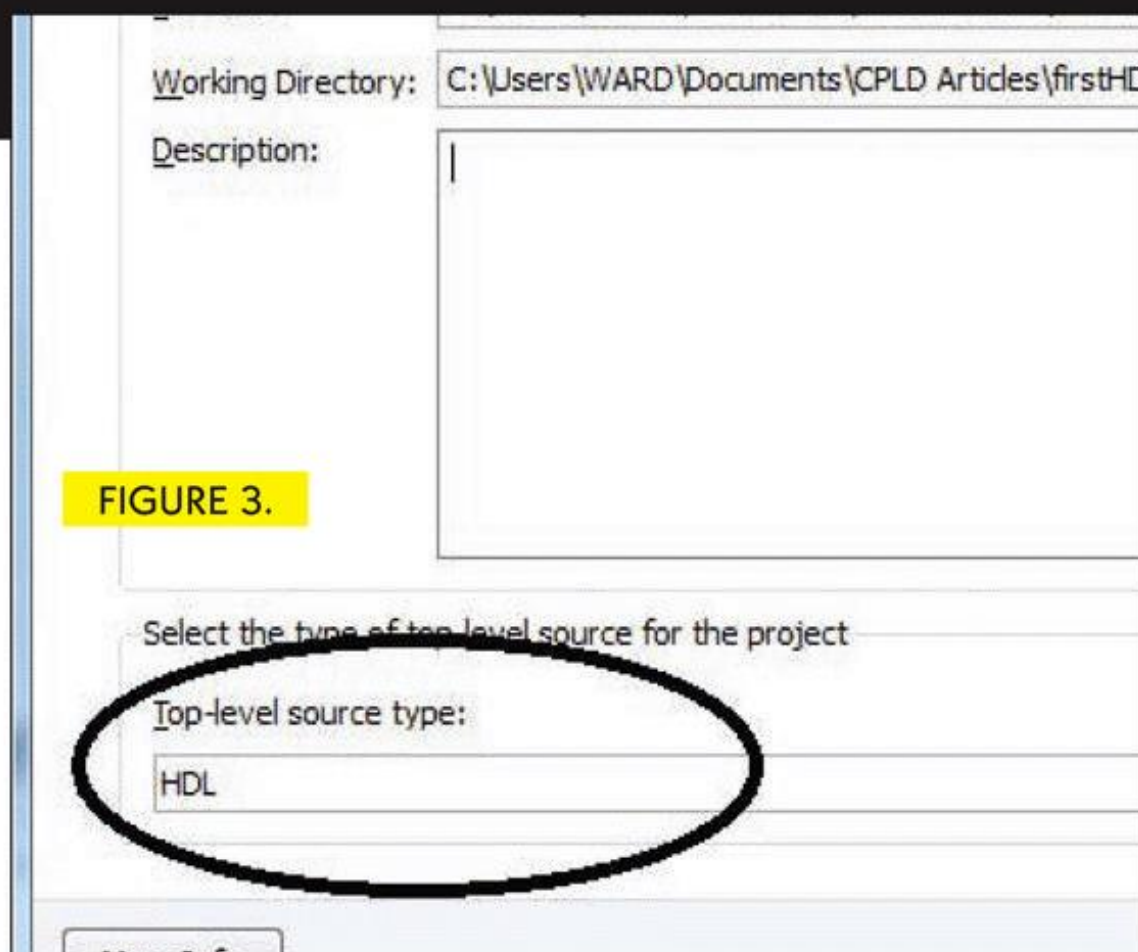


FIGURE 3.

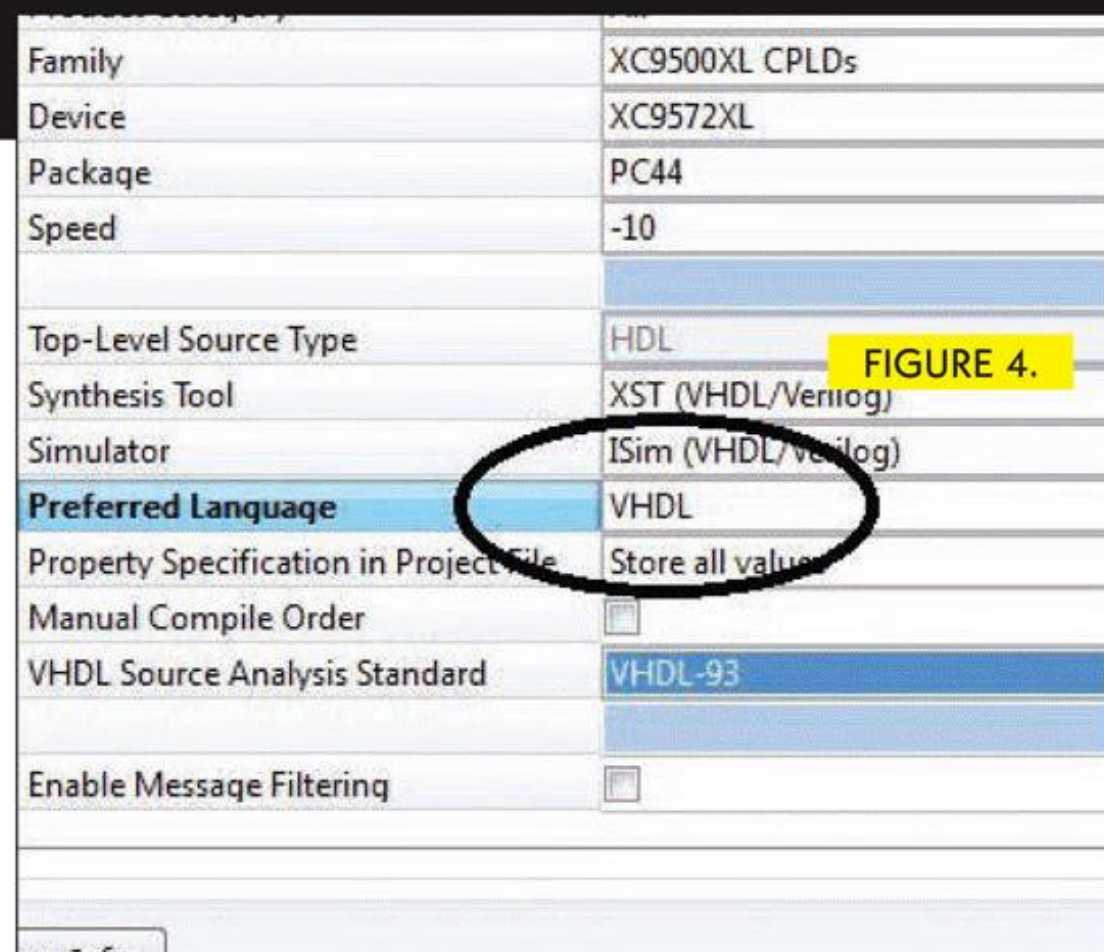


FIGURE 4.

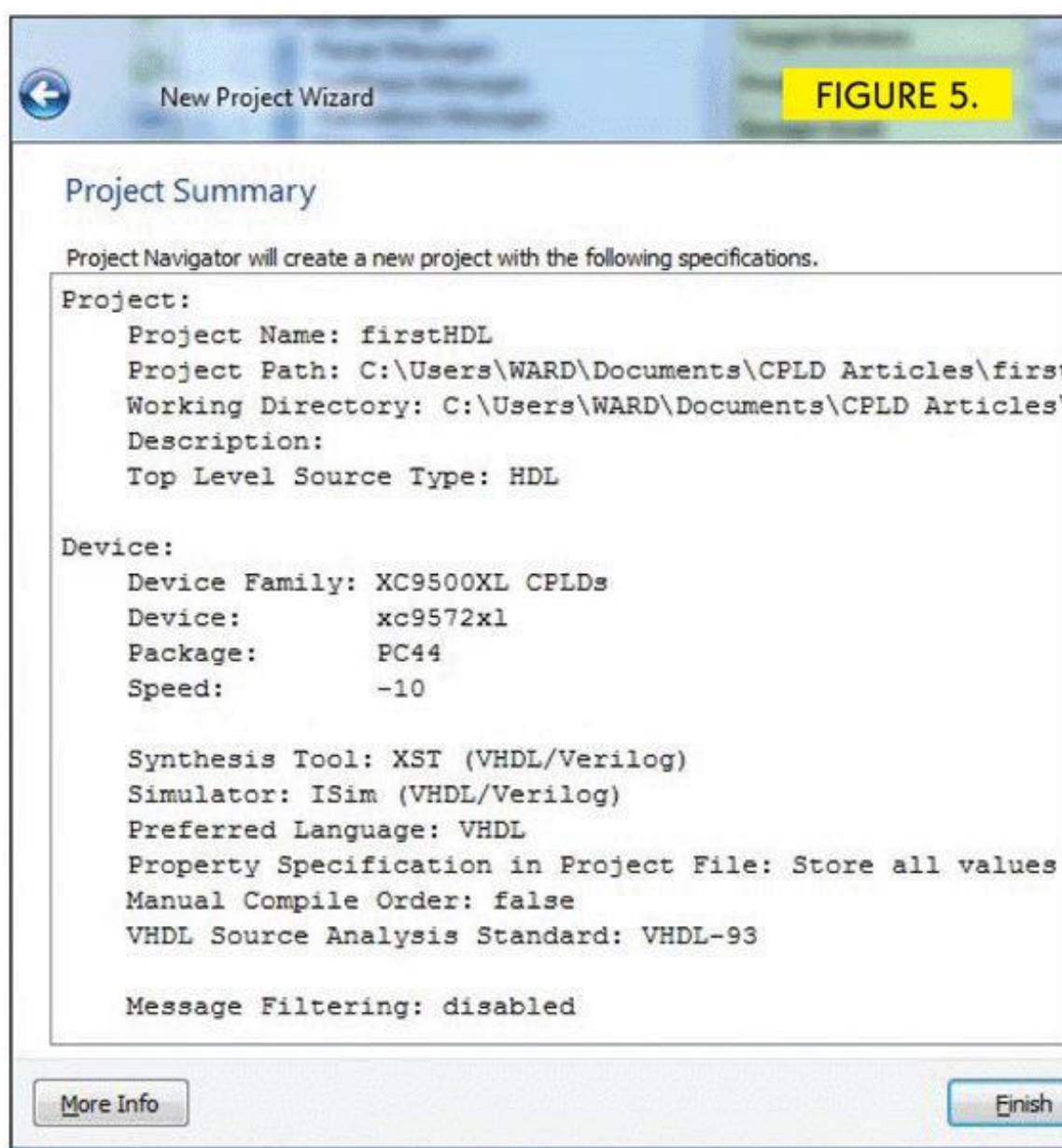


FIGURE 5.

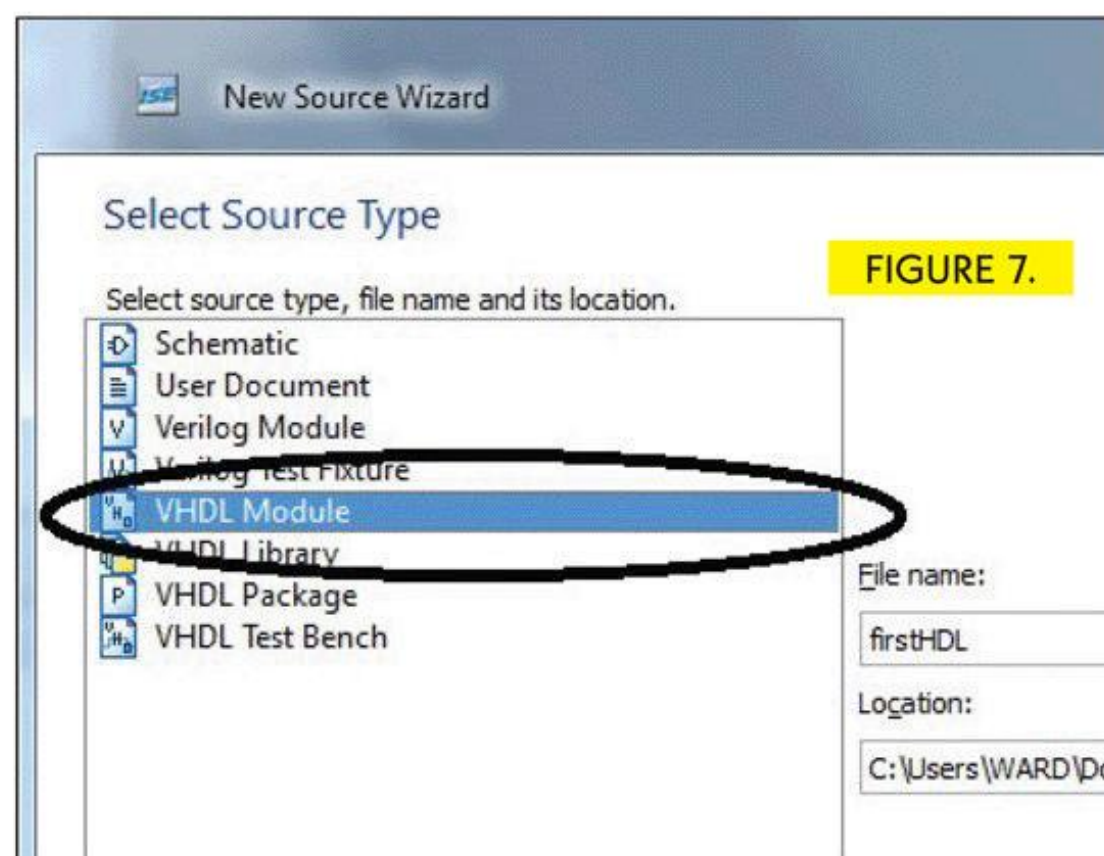


FIGURE 7.

Navigator. First, open the Xilinx program and from the start tab, select new project (see **Figure 2**). From the Create New Project window, locate and name your project, and at the bottom select HDL as the top-level source (see **Figure 3**). In the Project Settings window, the CPLD information should be the same as in the past projects, however, make

sure that the preferred language is set as VHDL (see **Figure 4**). Select finish from the Project Summary window as shown in **Figure 5**. You will now be taken back to the main ISE Project Navigator window. From here, select Project > New Source, as in **Figure 6**. From the Select Source Type window, select VHDL Module and give your file a name and location (see **Figure 7**).

In the Define Module window, you can enter your port names and whether they're inputs or outputs. Or, you can simply click on next and fill this information into the entity section later (see **Figure 8**).

Figure 9 is a summary window; select finish from here. You will now be taken into the HDL template that was created for

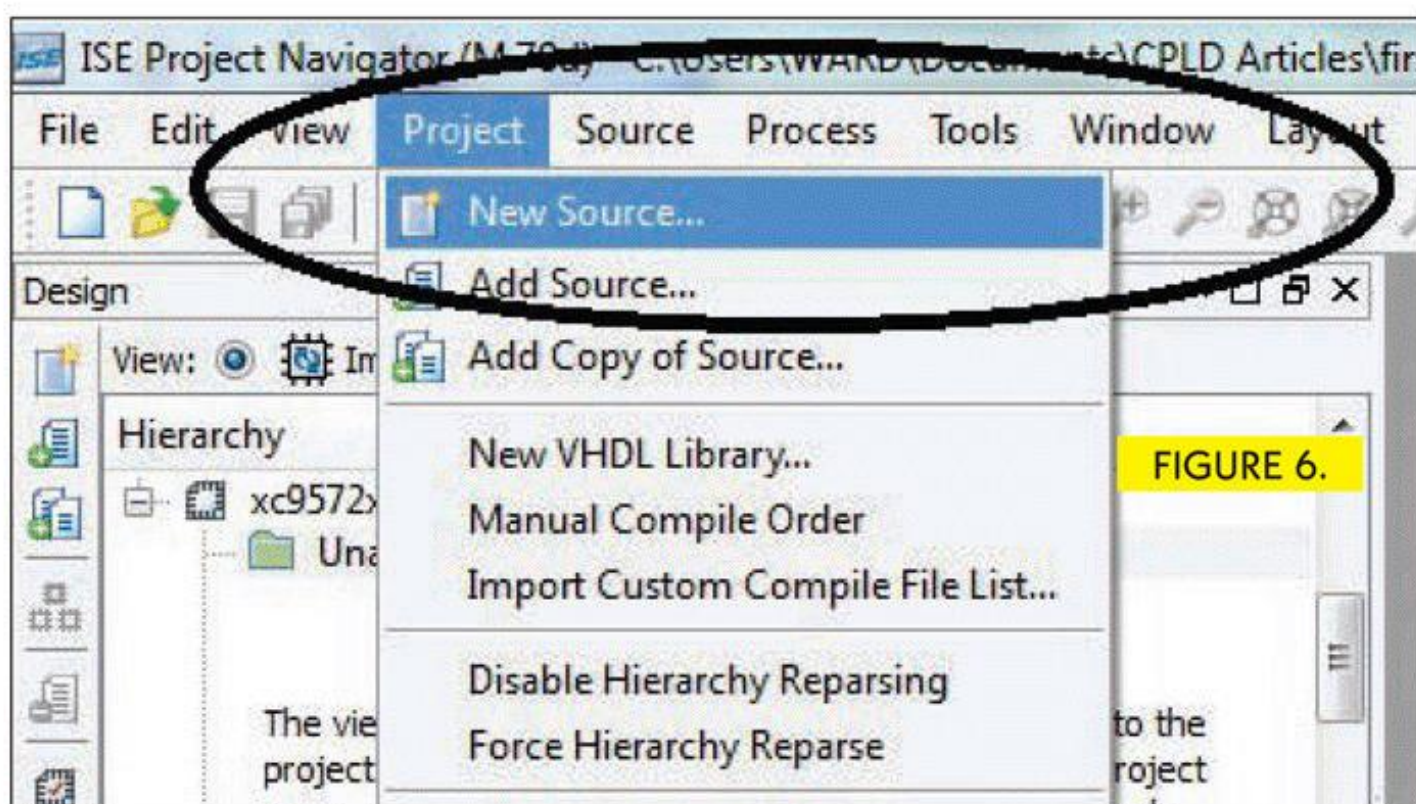


FIGURE 6.

New Source Wizard

Define Module

Specify ports for module.

Entity name: firstHDL

Architecture name: Behavioral

Port Name	Direction	Bus
A	in	☐
B	in	☐
X	out	☐
	in	☐
	in	☐
	in	☐
	in	☐

FIGURE 8.

Summary

Project Navigator will create a new skeleton source with the following specifications:

Add to Project: Yes
 Source Directory: C:\Users\WARD\Documents\CPLD Articles\firstHDL
 Source Type: VHDL Module
 Source Name: firstHDL.vhd

Entity name: firstHDL
 Architecture name: Behavioral

Port Definitions:

Port Name	Pin	Direction
A	Pin	in
B	Pin	in
X	Pin	out

FIGURE 9.

```

18  --
19  -----
20  library IEEE;
21  use IEEE.STD_LOGIC_1164.ALL;
22
23  -- Uncomment the following library declaration
24  -- arithmetic functions with Signed or Unsigned
25  --use IEEE.NUMERIC_STD.ALL;
26
27  -- Uncomment the following library declaration
28  -- any Xilinx primitives in this code.
29  --library UNISIM;
30  --use UNISIM.VComponents.all;
31
32  entity firstHDL is
33      Port ( A : in  STD_LOGIC;
34            B : in  STD_LOGIC;
35            X : out STD_LOGIC);
36  end firstHDL;
37
38  architecture Behavioral of firstHDL is
39
40  begin
41
42
43  end Behavioral;
44
45
  
```

FIGURE 10.

you (Figure 10).

Notice in the HDL template that the entity section is already completed for you, if you entered port names and directions from the earlier Define Module window; refer back to Figure 8. Notice also there are several lines of comments following two dashes which are green in color. The only real code or instructions that the compiler will use — other than the entity and architecture sections — are the library and use instructions on lines 20 and 21. The architecture section, however, is not completed for you.

Figure 11 shows the HDL code for the AND gate entered into the architecture section. From here on, everything is accomplished the same as it was when entering a schematic circuit. Select Implement Top Module (the green play icon) and if it compiles okay, you can go on into the Impact program and program your CPLD. If there are errors in your source code, error messages will be displayed in the console window at the bottom of the Navigator screen showing you which lines had problems. We won't go into all of the details how this is done here; you can refer back to Part 2 to see all of the steps to go through to compile and program a CPLD.

```

34      B : in  STD_LOGIC;
35      X : out STD_LOGIC);
36  end firstHDL;
37
38  architecture Behavioral of firstHDL is
39
40  begin
41
42      X <= A AND B;
43
44  end Behavioral;
45
  
```

FIGURE 11.

```

33  PORT(A, B :IN STD_LOGIC;
34        Q, R, S, T, U, V, W :OUT STD_LOGIC);
35  end HDL_3;
36
37  architecture Behavioral of HDL_3 is
38
39  begin
40      Q <= A AND B;
41      R <= A OR B;
42      S <= A NAND B;
43      T <= A NOR B;
44      U <= A XOR B;
45      V <= A XNOR B;
46      W <= NOT A;
47
48  end Behavioral;
  
```

FIGURE 12.


```

32 entity TRUTH_TABLE_1 is
33     PORT(A, B, C :IN STD_LOGIC;  --truth table set-up 3 inputs 1 output
34           X :OUT STD_LOGIC);
35 end TRUTH_TABLE_1;
36
37 architecture Behavioral of TRUTH_TABLE_1 is
38     SIGNAL BITS_IN :STD_LOGIC_VECTOR(0 TO 2);
39 begin
40     BITS_IN <= A & B & C;  --concatenate the 3 inputs
41
42     WITH BITS_IN SELECT
43         X <= '0' WHEN "000",
44              '1' WHEN "001",
45              '0' WHEN "010",
46              '0' WHEN "011",
47              '0' WHEN "100",
48              '0' WHEN "101",
49              '0' WHEN "110",
49              '0' WHEN "111";
49 end Behavioral;

```

FIGURE 13.

```

32 entity TRUTH_TABLE_2 is
33     PORT(A, B, C :IN STD_LOGIC;  --3 inputs
34           DISPLAY :OUT INTEGER RANGE 0 TO 7);  --this will generate 3 outp
35 end TRUTH_TABLE_2;
36
37 architecture Behavioral of TRUTH_TABLE_2 is
38     SIGNAL BITS_IN :STD_LOGIC_VECTOR(0 TO 2);
39 begin
40     BITS_IN <= A & B & C;  --concatenate the 3 inputs together
41
42     WITH BITS_IN SELECT
43         DISPLAY <= 0 WHEN "000",
44                   1 WHEN "001",
45                   2 WHEN "010",
46                   3 WHEN "011",
47                   4 WHEN "100",
48                   5 WHEN "101",
49                   6 WHEN "110",
50                   7 WHEN "111";

```

FIGURE 14.

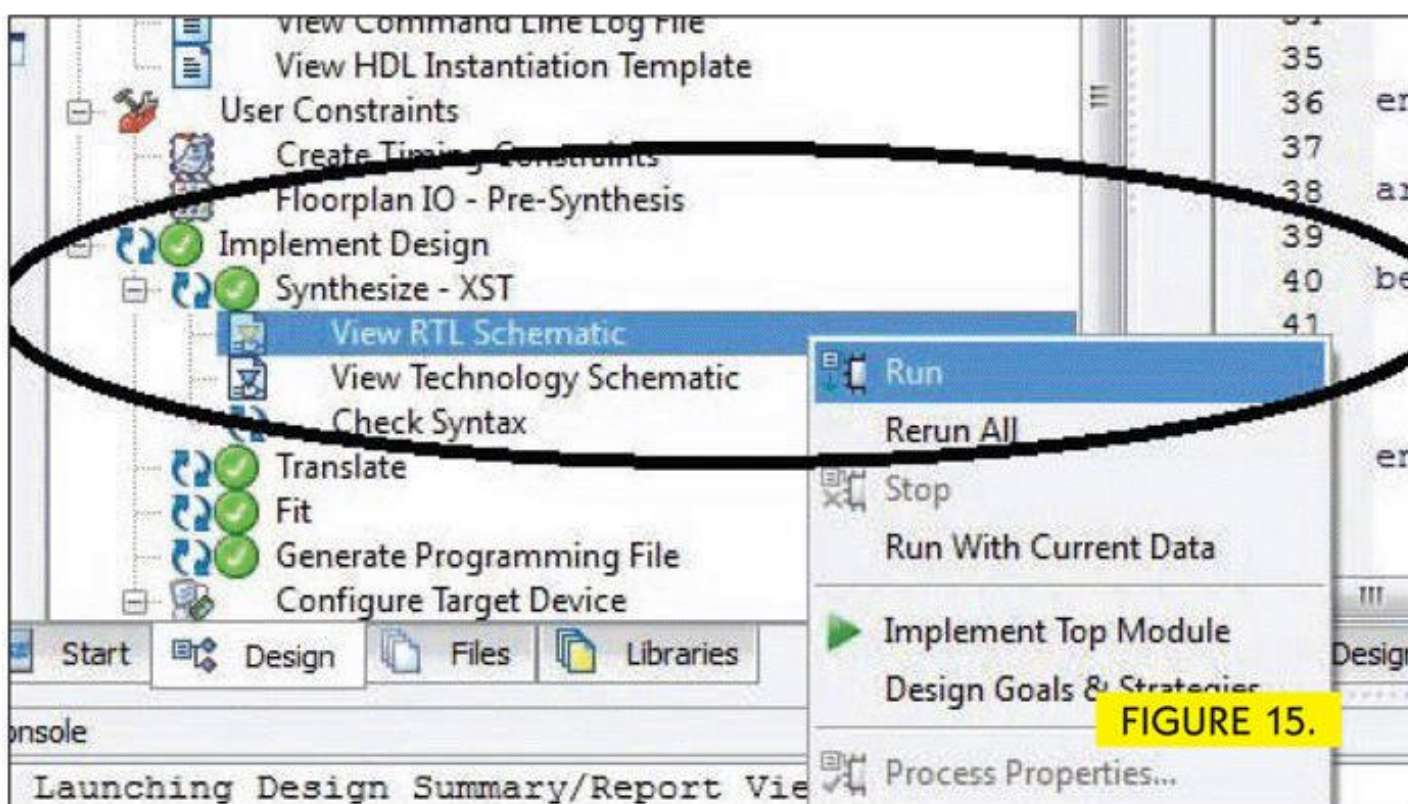


FIGURE 15.

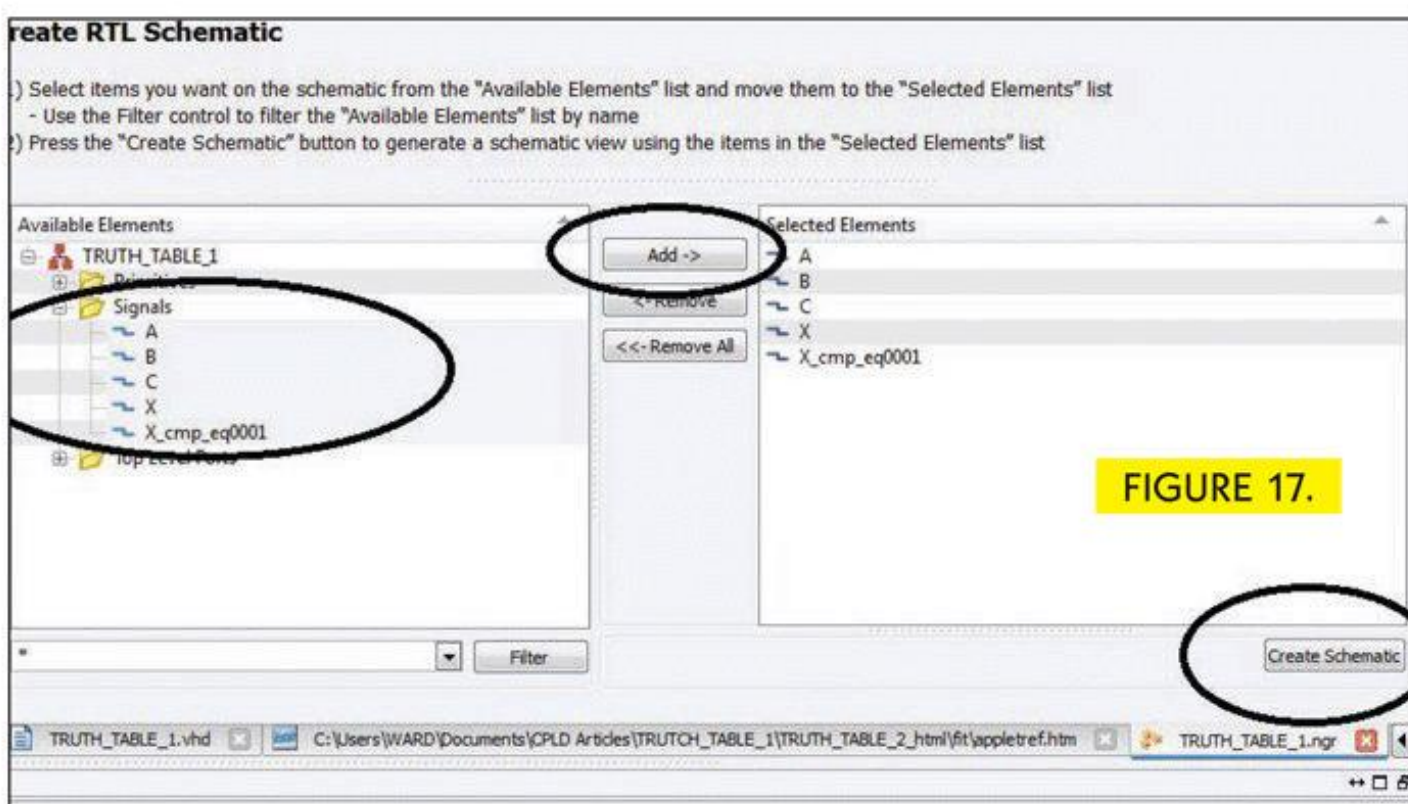


FIGURE 17.

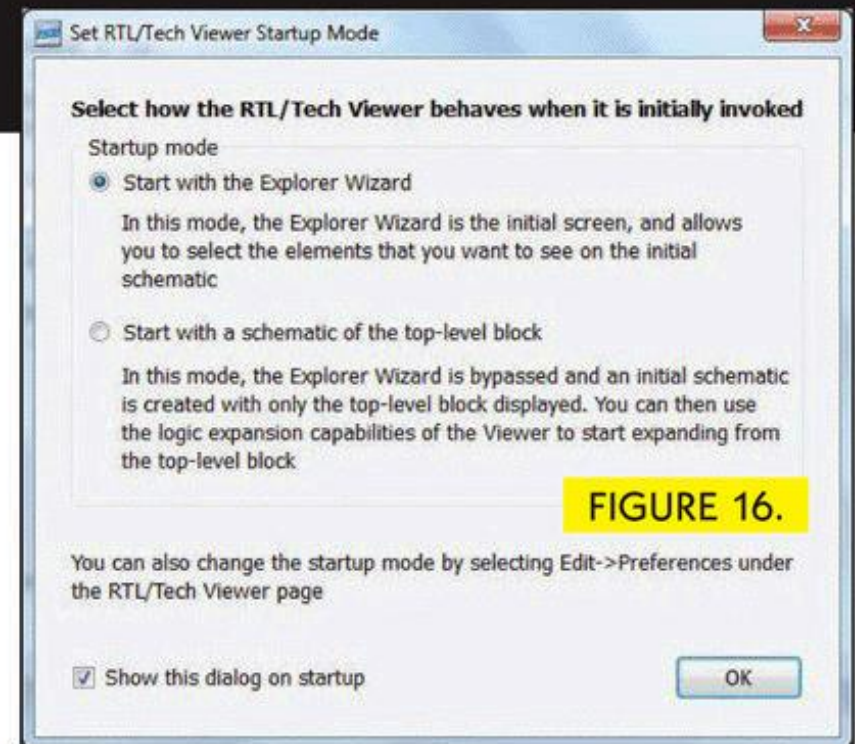


FIGURE 16.

All of the other basic logic gate functions can be entered in the same way the AND gate was; refer to **Figure 12**.

One of the most common methods used to plan and design digital logic circuits is to use a truth table. Let's look at one way to enter truth table information into HDL code, (**Figure 13**). Let's look at what is taking place in this HDL listing. Line #32 begins the entity section. Line #33 defines three inputs named A, B, and C, that are defined as STD_LOGIC types of inputs. Line #33 also illustrates how comments can be added after the HDL code using the two dashes.

The inputs could also have been declared as BIT types ":IN BIT." A BIT type can either be a '1' or a '0', and nothing else. A STD_LOGIC type can be a '1' or a '0' or several other things, as well. A STD_LOGIC can also be a 'Z' (high impedance; a '-' (don't care); an 'X' (unknown); an 'L' (weak '0'), or an 'H' (weak '1'). When the Xilinx program sets up your HDL template, it usually defines the pins as STD_LOGIC. Line #34 declares an output named "X" as an STD_LOGIC type.

Line #35 ends the entity section. Line #36 is white space which is ignored by the compiler. Line #37 begins the architecture section. Line #38 defines a locally used SIGNAL named BITS_IN which will only be used in the architecture section. It sets up a three-bit STD_LOGIC_VECTOR which will be used to hold A, B, and C so they can be evaluated together rather than one at a time. Line #39 is the begin label which is necessary inside the architecture section.

Line #40 concatenates the three inputs A, B, and C, and puts them into BITS_IN so they can be evaluated together. Line #41 is white space. Line #42 sets up the WITH SELECT WHEN structure. Line #43 will make the output or X a 0 when the three inputs

are all 0. Lines #44 through #46 list the other conditions of the truth table and what X will be with each condition of A, B, and C. By the way, the order of A, B, and C in relation to the three 0s to the right of the word WHEN is the same. That is, A is the left 0, B is the center 0, and C is the right 0.

Line #47 "WHEN OTHERS" can be used if you want all of the other truth table outputs to be the same and do not wish to list every possible input condition. Note, however, that the WHEN OTHERS statement is still necessary even if you do list all of the possible input combinations. Line #48 ends the architecture section.

Notice the use of single quotes for the ones and zeroes before the WHEN, and the use of the double quotes after the WHEN in lines #43 through #47. Notice also the use of a comma after lines #43 through #46 and a semi-colon at the end of line #47. Not having these commas, quotes, and semi-colons in the right places will cause errors when attempting to compile the code.

Let's now look at entering a truth table that has three inputs and three outputs into HDL. Check out **Figure 14**. Notice in line #34 there will be an output named DISPLAY that has an integer range of 0 to 7. This will automatically set up three output bits named DISPLAY<0>, DISPLAY<1>, and DISPLAY<2>. If you put other numbers in the range such as 0 to 10, for example, the compiler will set up four outputs since it will take four bits to place values up to 10 out of the circuit. Notice in lines #43 through #51 that the output values are decimal and are not enclosed in any type of quotes.

Other than that, the code listing looks the same as the single output code shown earlier. Of course, you can do many more things with HDL than just these two truth table examples. Being able to enter truth table information into HDL will get you a solid start into using HDL.

Let's look at a couple of other things that the Xilinx software can do at this time. One thing you can do is generate a schematic diagram from HDL code and vice versa. To generate a schematic from HDL code, select Implement Design > Synthesize – XST > View RTL Schematic > left-click then right-click, and from the drop-down menu select Run (see **Figure 15**). From the next window, make sure the Start the Explorer Wizard radio button is selected and then select OK (see **Figure 16**). From the Create RTL Schematic window, select all signals > Add > and then Create Schematic, as shown in **Figure 17**. You will now see the schematic diagram that was generated from your HDL source code in **Figure 18**. To turn a schematic

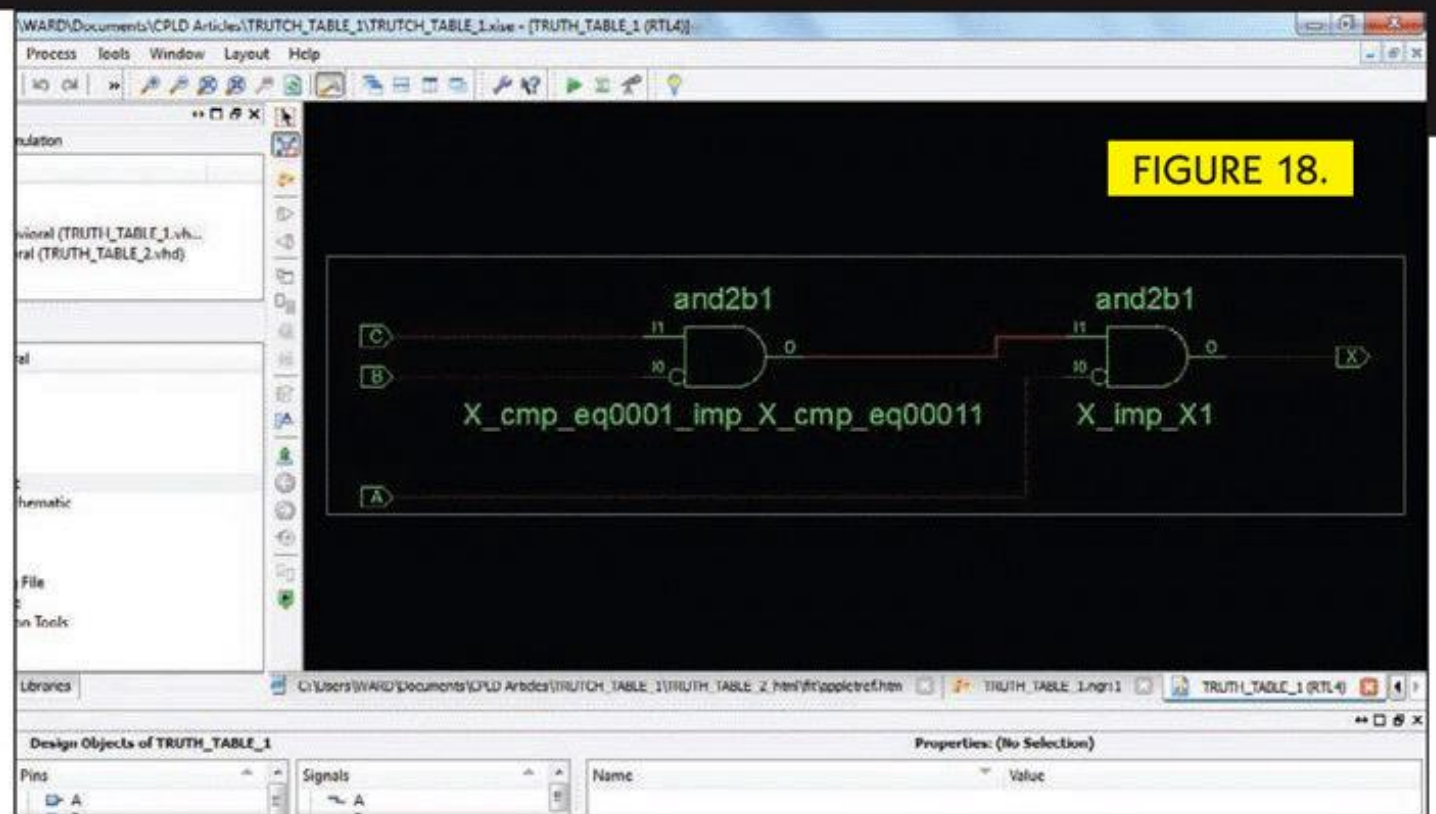


FIGURE 18.

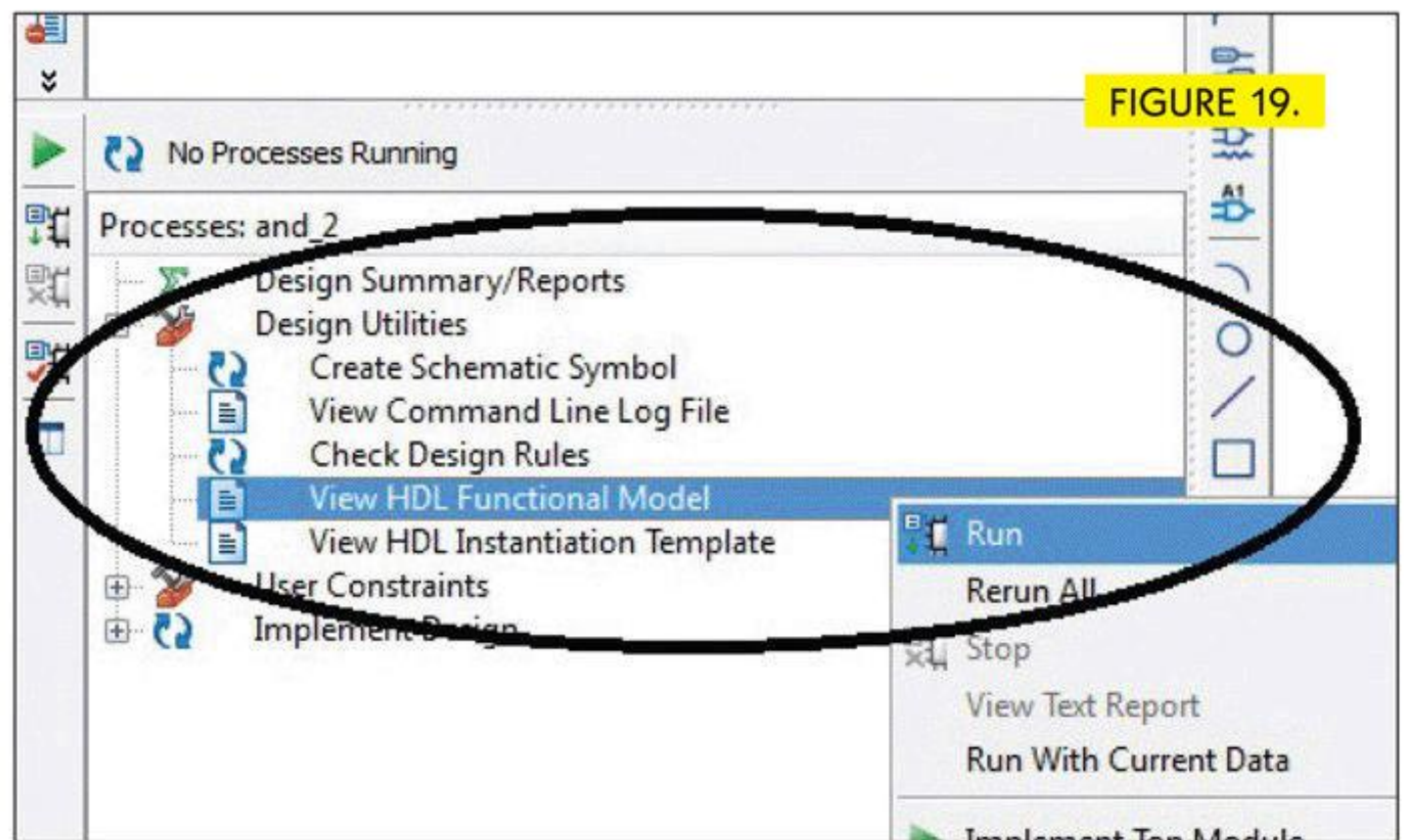


FIGURE 19.

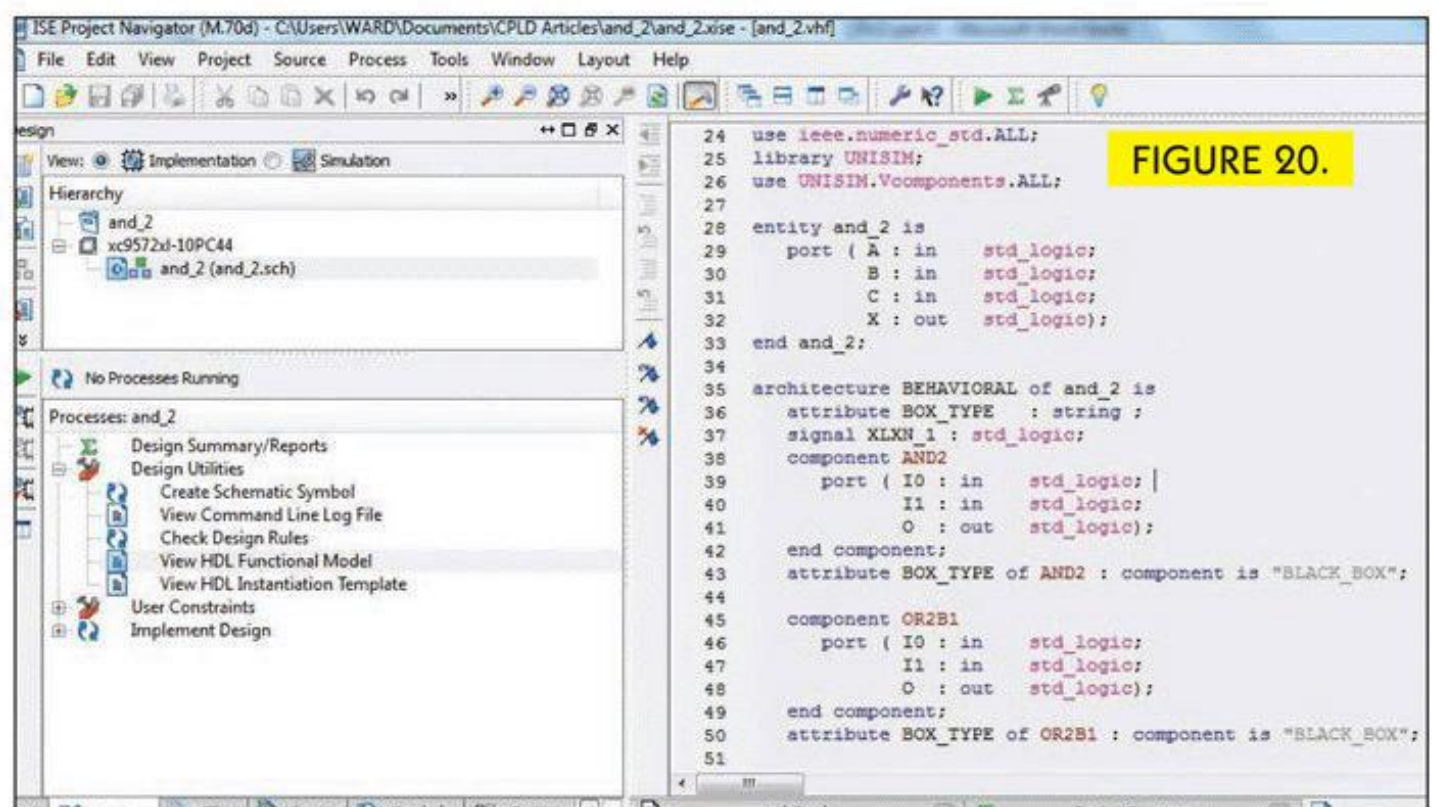


FIGURE 20.

diagram into an HDL code listing, select Design Utilities > View HDL Functional Model > left-click then right-click, and from the drop-down menu select Run (see **Figure 19**). You will now see the HDL code listing generated from that schematic diagram (**Figure 20**).

The final article next month will use a CPLD to control a mobile robotics platform to follow a dark tape line on a table-top. This will tie all of the principles that have been introduced in these articles together. **SV**

Make Your Robot's Wires Extinct

by Fred Eady

Way back when dinosaurs roamed the hills of Tennessee, I was learning how to wire up tube and transistor circuits. There was no such thing as ExpressPCB. In fact, there was no such thing as a BBS or the Internet. Only the professional magazine writers of the day with access to a printed circuit board house could muster an article that included a project mounted on a fancy copper clad piece of fiberglass. Meanwhile, I continued to string wire between circuit points and electrically seal them with solder.

Years passed. The dinosaurs died. Dial-up BBS sites gave way to the Internet and professional printed circuit board houses reached out to folks like you and me. Telephones lost their wires and personal computer Ethernet cables disappeared. I'll bet you're still wired to that microcontroller that you're going to use to control that rolling aluminum slab you call a robot. Read on and I'll show you how to lose the wires.

www.servomagazine.com/index.php?/magazine/article/june2011_Eady

A New 8051-based Powerhouse

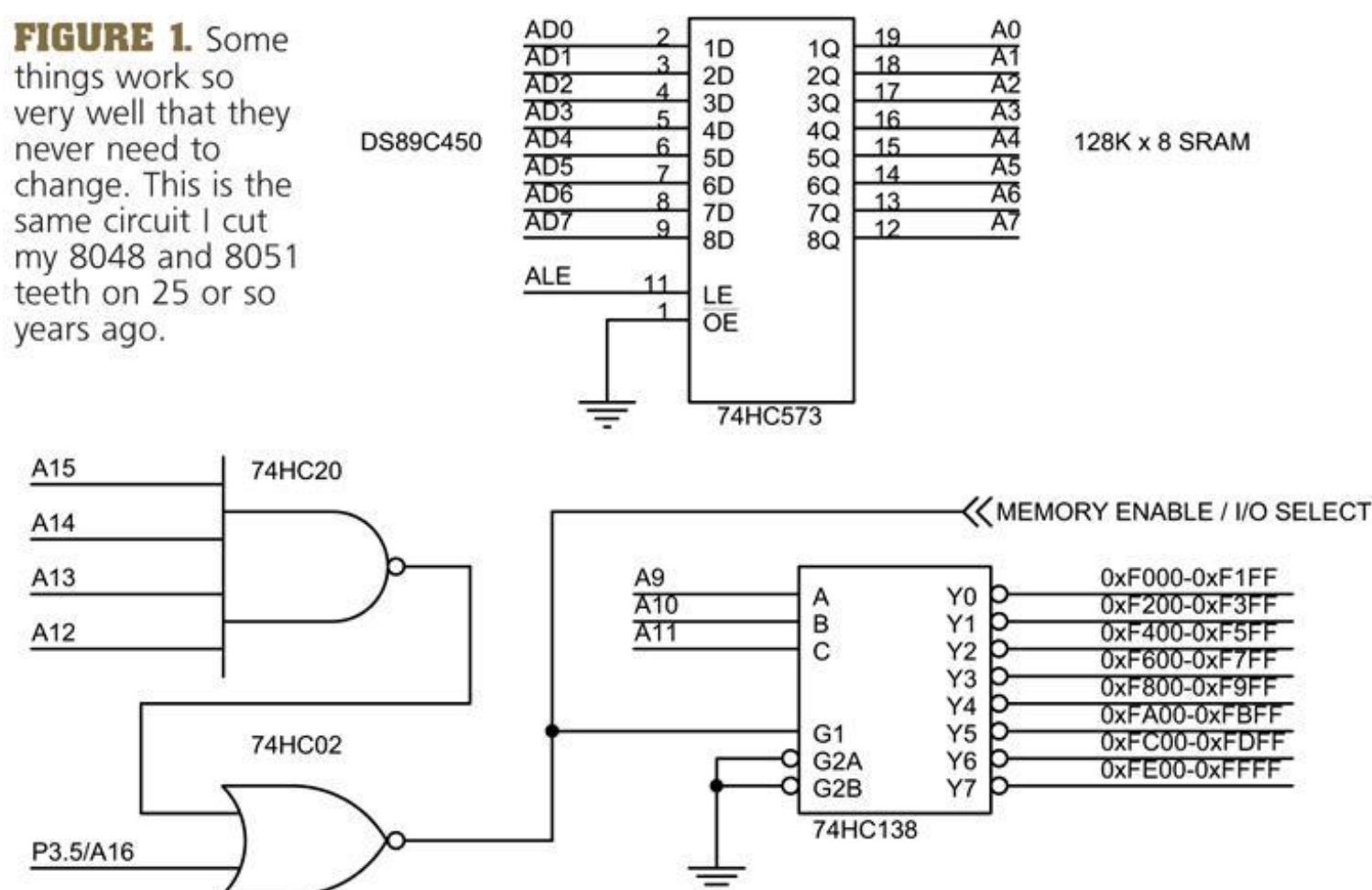
Before we put on our pointy hats and discuss the radios, I'm going to introduce you to the SPECTRUM ACE 2a — a really neat, self-contained computing platform from I2I Controls. ACE is short for Advanced Control Environment. The SPECTRUM ACE 2a single board computer is going to be right down your alley. As a robot

head, you are ALL about control, and that's what the SPECTRUM ACE 2a is designed to do.

The ACE 2a is the largest microcontroller platform in the SPECTRUM ACE family. The ACE 2a is based on the Dallas Semiconductor DS89C450 ultra high speed Flash microcontroller. If you roamed with the dinosaurs, you know all about the 8051 microcontroller and its variants. The DS89C450 microcontroller does in one clock cycle what the T-Rex 8051 does in 12 clock cycles.

Remember the 8751? The 8751 had EPROM under quartz that could be electrically programmed and erased via a UV lamp. If you had a few days between spins, you could set your programmed 8751 in the sunlight and eventually somewhat erase the program memory area. I used to keep opaque tabs to cover the quartz window to prevent accidental erasure of my 8751 code that took days (and sometimes weeks) to perfect. Not so with the DS89C450 microcontroller. Like just about every modern microcontroller, the DS89C450 keeps your magic potion safe in 32K of eight-bit Flash. The DS89C450 doesn't stray far from the things that make it a true

FIGURE 1. Some things work so very well that they never need to change. This is the same circuit I cut my 8048 and 8051 teeth on 25 or so years ago.



8051 descendent. For instance, the classic 74HC573 data latch and 74HC138 memory decoder circuitry you see in **Figure 1** is utilized in the SPECTRUM ACE 2a hardware. In the old days, we had to hang our external SRAM out on these address markers as the stock 8051 only supported 128 bytes of SRAM on chip. I recall that the 2K x 8 6116 SRAM was one of my favorites. The ACE 2a still hangs SRAM out on the address decoders. However, instead of a paltry 2K of SRAM, the ACE 2a puts clothespins on 128K of eight-bit SRAM.

Every well-heeled robot can deduce the time of day. Thus, any cousin of Robby is endowed with an electronic clock of some kind. The ACE 2a design includes a Dallas Semiconductor DS28DG02 real time clock. In addition to providing a battery-backed RTCC function, the real time clock also brings some general-purpose I/O lines to the party.

Although you may think digital rules, we actually live in an analog world. To help you and your intelligent bucket of bolts cope, the ACE 2a is equipped with a Dallas Semiconductor 1-Wire DS2450S 1-Wire quad analog-to-digital converter (ADC).

Stepping back and pondering the power of that collection of electronic components under the Canon in **Photo 1**, you realize that the ACE 2a gives you full access to the CPU data bus, the low order address bus, and an octet of 512-byte memory-mapped I/O chip selects spanning a 4K block. If we were to total up these goodies, the list would include:

- DS2450S 1-Wire A-to-D
- Six Hardware Interrupts
- Two Full-Duplex 115K baud UARTs
- One PWM (Software Generated)
- 10 General-Purpose I/O Lines
- 32K of eight-bit Flash
- 128K of eight-bit SRAM
- Reset Button
- +5 VDC Power Supply

As my mom would say in a slurpy Southern drawl, "That's nice." What she really means is, "So What?" Somebody's got to write drivers for all of those "features." Usually, that somebody is you.

Not this time. The console portal of the ACE 2a provides you access to ALEC. ALEC is smart, but it is not of our world. ALEC is short for Advanced Language for Embedded Control. ALEC is SPECTRUM ACE 2a standard equipment. Everything necessary to implement and drive that aforementioned list of features is part of ALEC. ALEC don't need no stinkin' compiler. ALEC don't need no stinkin' hardware programmer. Your programs can be written, edited, debugged, and executed via ALEC and a simple RS-232 portal, which is part of the SPECTRUM ACE 2a.

TX, RX, and GND in an Ethernet Sort of Way

There are a number of radio modules that we could

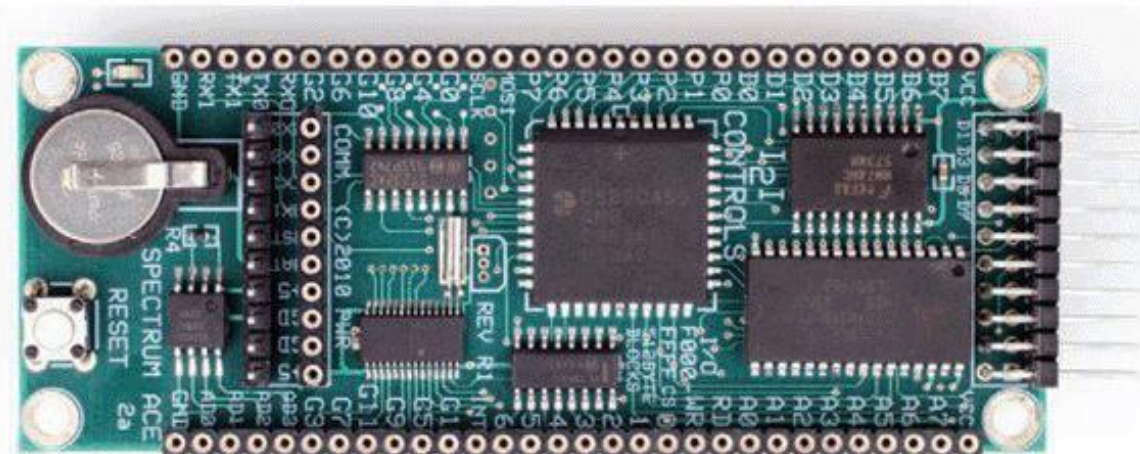


PHOTO 1. Back then, I would have slain old T-Rex myself to have access to this technology. Programming an 8051 is like playing a Fender Telecaster through a Fender tube amp. Good vibes.

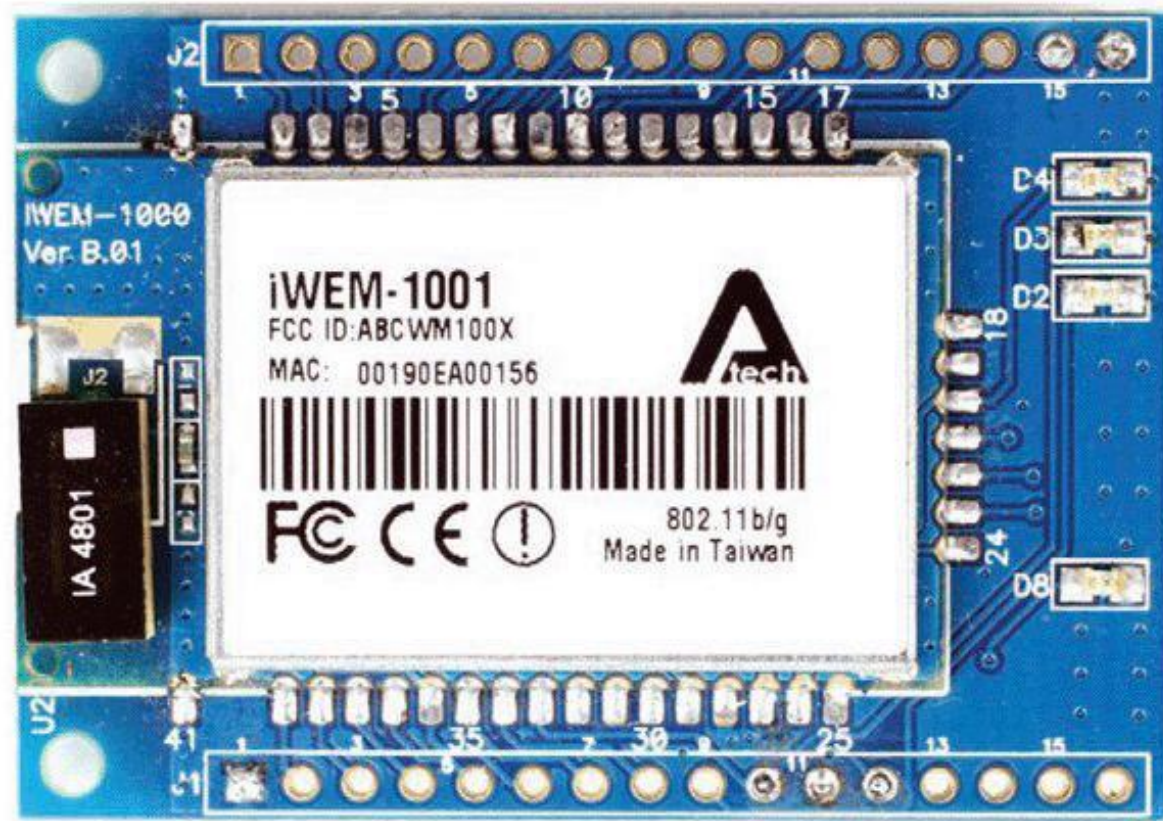


PHOTO 2. The iWEM-1001 is an ultra low power consumption 802.11b/g Wi-Fi module that sports a basic three-wire UART interface.

use to eliminate that wire that connects your microcontroller's UART to a console. However, we're going to cut the wire in such a way as to allow you to access your mechanical animal using Internet protocols that look like a standard three-wire RS-232 hookup.

Photo 2 is a top-side shot of the Atech iWEM-1001 ultra low power wireless embedded module. The iWEM-1001 is an 802.11b/g device that can operate at 54 Mbps. Our iWEM-1001 is equipped with an embedded chip antenna. It can also be had with a standard UFL/SMA antenna connector. If your microcontroller has a UART, you're ready to rock as the iWEM-1001 only requires a simple three-wire serial connection. It supports standard baud rates between 2400 bps and 115200 bps.

The iWEM-1001 is actually a fully functional Wi-Fi radio station. Its chip antenna feeds an embedded 32-bit CPU that is running an operating system and a network stack. A sequence of "+++" sent from a terminal emulator such as HyperTerminal or Tera Term Pro forces the iWEM-1001 from data mode to command mode. Once in command mode, the iWEM-1001's configuration can be viewed or set. Data mode exposes its business end. Sending the word "exit" to the iWEM-1001 module via the terminal emulator terminates command mode and forces its module into data mode. When the iWEM-1001 is powered up, it enters data

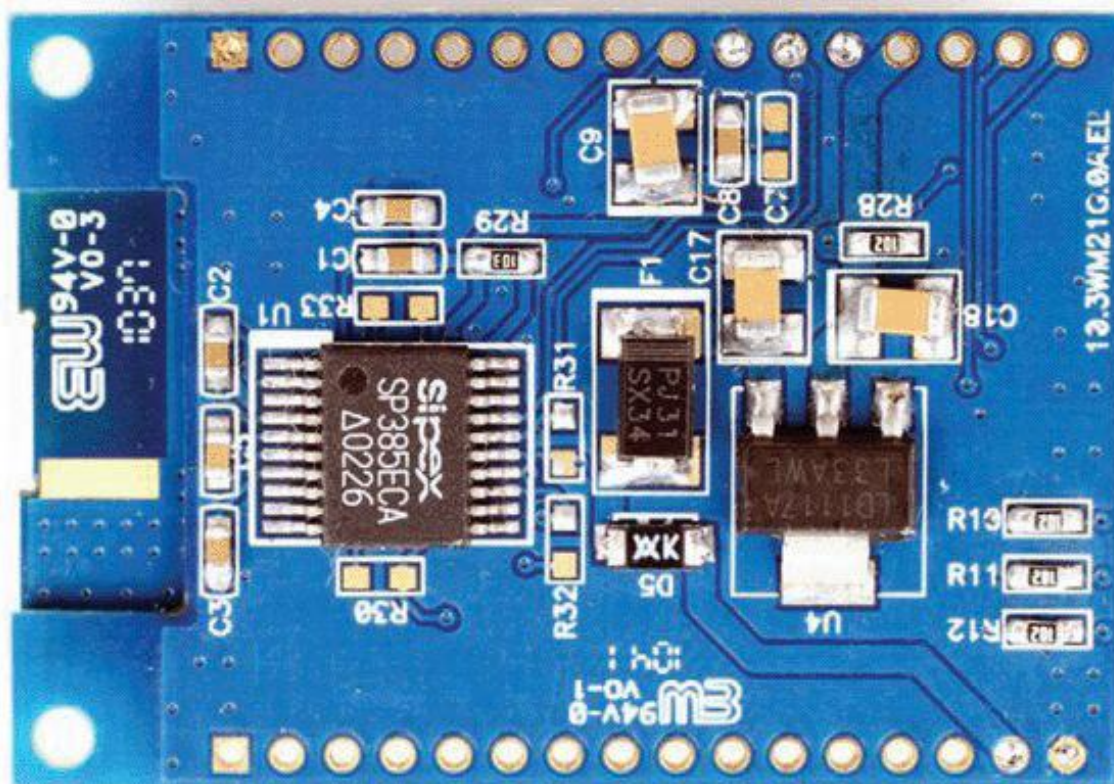


PHOTO 3. What you see here is the iWEM-1001's LD1117A 3.3 volt LDO voltage regulator circuitry and a SP385E true +3V or +5.0 volt RS-232 line driver/receiver.

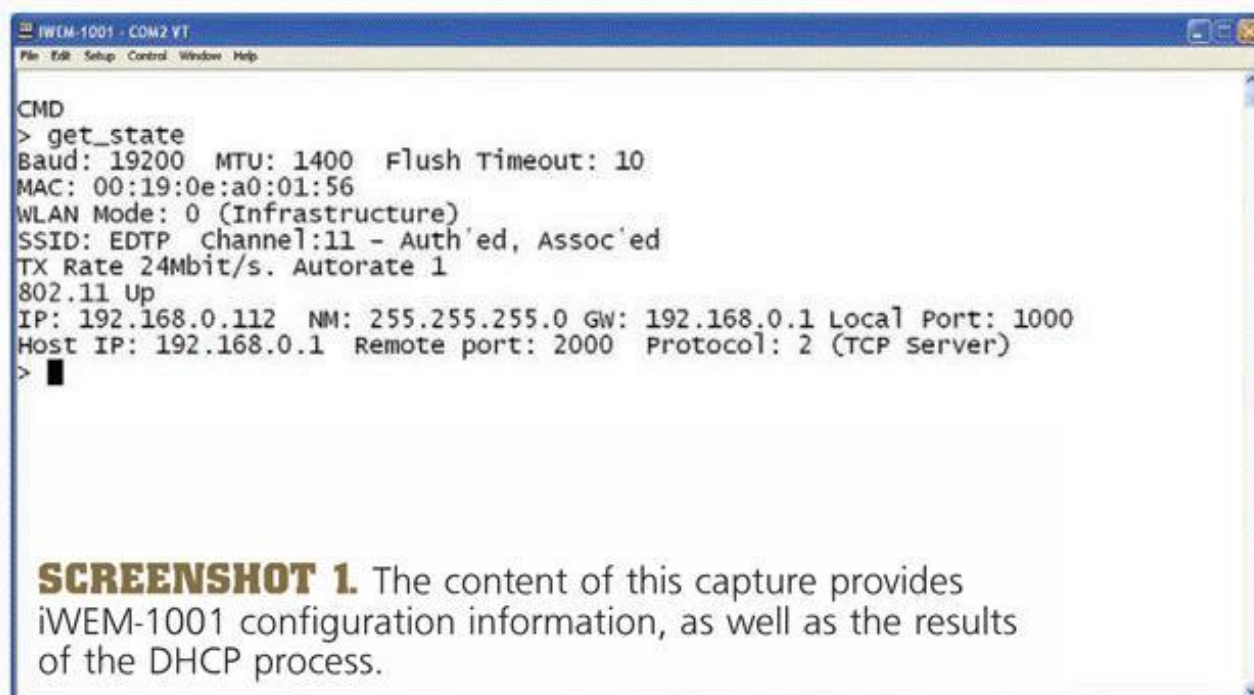
mode by default. If configured to do so, as soon as its internal CPU boots the module attempts to associate with a WLAN network. The iWEM-1001 can associate via DHCP or static network addressing. Once it finds a friendly WLAN, it can converse with other network nodes via TCP/IP, UDP, ARP, and ICMP as a client or server.

The serial side of our three-wire to Internet protocol bridge is facing the camera lens in **Photo 3**.

A Sipex 385E RS-232 line driver/receiver acts as the gateway to the iWEM-1001's Internet protocol engine. The Sipex 385E can operate with power supplies of either +3 volts or +5 volts. As far as power supply voltage is concerned, the iWEM-1001's operational range is 2.4 VDC to 3.7 VDC. So, an LD1117AL33 LDO voltage regulator with a fixed output voltage of 3.3 VDC is holding court on the serial side of the iWEM-1001.

Setting Up the iWEM-1001

The iWEM-1001 command set consists of three command types. Action commands execute network functions. For instance, an action command exists for scanning, connecting, and disconnecting from the network. Configuration commands ultimately load the iWEM-1001



SCREENSHOT 1. The content of this capture provides iWEM-1001 configuration information, as well as the results of the DHCP process.

and network configuration information into the iWEM-1001's Flash memory. Configuration information must be saved using the `save` command in order to be updated and used following the next iWEM-1001 boot. Configuration commands act on the iWEM-1001 hardware and the network. For example, the `set_baud_rate 19200` command sets the iWEM-1001 baud rate to 19200 bps. The network command `set_protocol 2` instructs the iWEM-1001 to act as a TCP server. Status commands allow the user to view the current state of the iWEM-1001's hardware and network settings. The status of the iWEM-1001 from a network point of view is also available by issuing the `get_state` command. The best way to get a grip on the iWEM-1001's command mode command set is to walk through setting up an iWEM-1001.

There are a few things we need to know before we can put the Wi-Fi side of the iWEM-1001 to work. Some of those things are obvious, such as the type of antenna it's fitted with. **Photo 2** tells us a chip antenna is installed. Issuing the `set_antenna 1` configuration command informs the iWEM-1001 that it will be receiving and transmitting using an embedded chip antenna. Here's some known information about the network our iWEM-1001 will participate in:

SSID	EDTP
802.11b/g Channel	11
Network Type	Infrastructure
WEP Key Number	1
WEP Key	112233445566778899AABBCCDD

We can use WLAN-oriented configuration commands to enter our network information:

```
set_ssid EDTP
set_channel 11
set_wlan_mode 0
set_passwd wep104 1 112233445566778899AABBCCDD
```

The SSID and channel match the associated configuration entries of the router that is in charge of our wireless network. At first glance, issuing the `set_wlan_mode 0` command is not necessary as our network type is infrastructure (uses an access point or router) and infrastructure (0) is the default network type. However, we would have to get up on our donkey and proclaim that this particular iWEM-1001 has never been configured before. I don't like riding the donkey. We'll make sure the iWEM-1001's network type is configured as infrastructure. If you're wondering where the WEP key number falls into all of this, it is the first argument of the `set_passwd wep104` command. If WEP is not your idea of security, the iWEM-1001 is also capable of securing the network using WPA (`set_passwd wpa <password>`). With our iWEM-1001 WLAN configuration complete, we can move on to the network configuration. Here's what we know:

DHCP	Enabled
Protocol	TCP Server

That doesn't seem like much, but using DHCP eliminates us from having to know the IP address particulars. The DHCP process will supply the iWEM-1001 with its IP address, the gateway IP address, and the netmask. The default TCP server local port is 1000. Since we aren't depending on another application to call the port numbers for us, the defaults for local and remote ports will work just fine here. So, we have a pair of network commands to enter:

```
set_dhcp 1
set_protocol 2
```

It would be nice if we initially configure the iWEM-1001 UART's baud rate to match the console speed of the SPECTRUM ACE 2a:

```
set_baud_rate 19200
```

That's all we need to do. There are a total of three UART configuration commands. We'll take the defaults on the packet size and flush timeout values as the default packet size is the maximum value of 1400, and the flush timeout value of 10 mS is plenty of cushion at 19200 bps. Save is an action command and if we want to keep our configuration entries, we had better issue it.

In that I was able to snap **Screenshot 1**, we definitely nailed the UART configuration. Let's peruse **Screenshot 1** and see how we did everywhere else. I tapped in "+++" into a 19200 bps Tera Term Pro session and as you can see, CMD and a command prompt were sent to the terminal emulator by the iWEM-1001 indicating that it has entered command mode. I issued the status command `get_state` to display the current thoughts of the iWEM-1001.

Recall that we configured the 19200 bps baud rate and left the MTU and flush timeout values to the discretion of the iWEM-1001 hardware. If you've read any of my Ethernet discussions or attended any of my Microchip MASTERS classes, you know that every Ethernet device that plans on participating in Internet activity must have a unique MAC address. Note that we did not have a configuration entry for a MAC address. That's because the MAC address was issued to Atech by the IEEE. We did have a say concerning the WLAN mode, as we chose to be safe rather than sitting on our donkey feeling sorry.

The SSID and channel were configured to make sure



SCREENSHOT 2. We're going to configure HyperTerminal to use an Ethernet interface instead of a serial interface.



SCREENSHOT 3. The combination of an IP address and port number results in a TCP socket. Our iWEM-1001 TCP socket happens to be a server socket that listens for incoming requests from clients.

that the iWEM-1001 associated with the router in the lab and one in a neighboring building. Everything went as planned as the iWEM-1001 was authenticated by our router and permitted to associate with the EDTP network. The TX

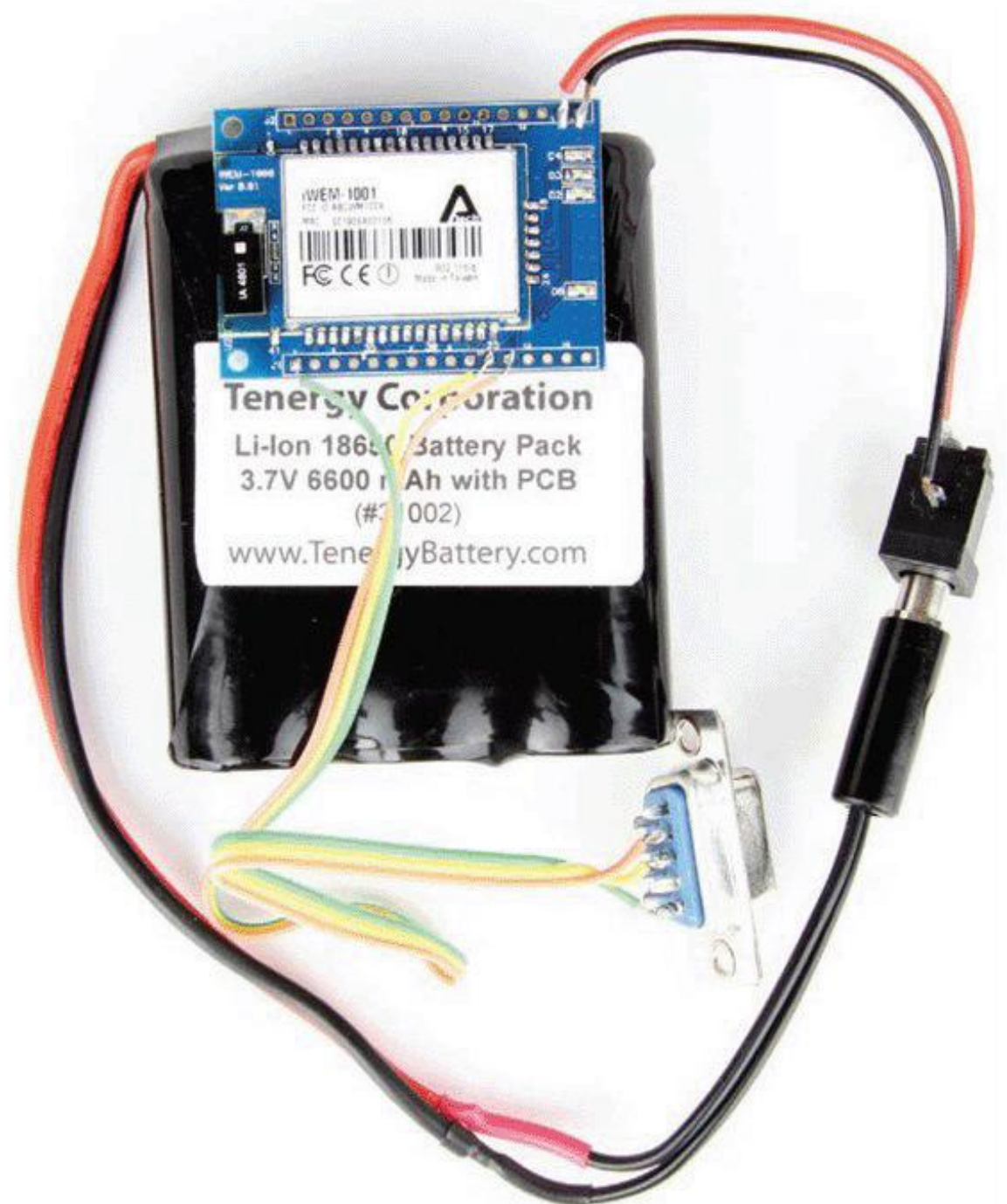
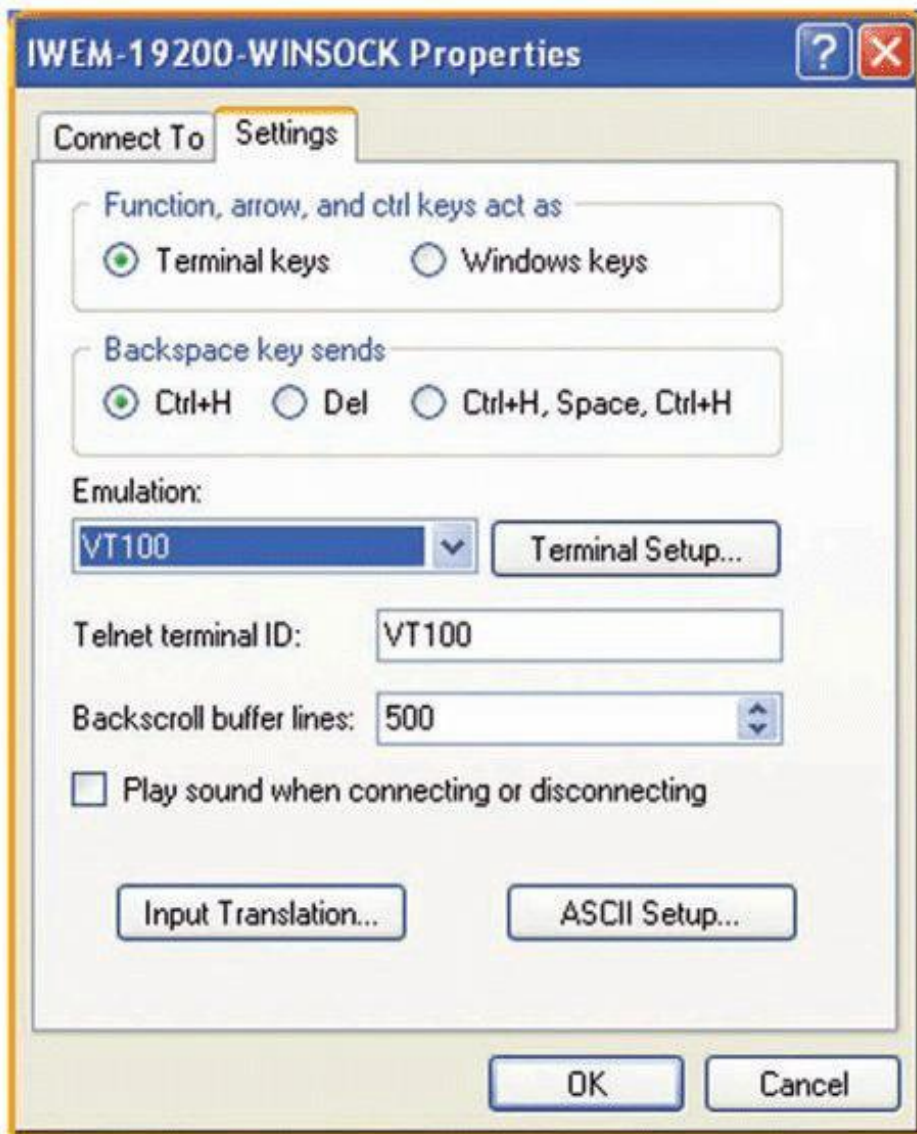
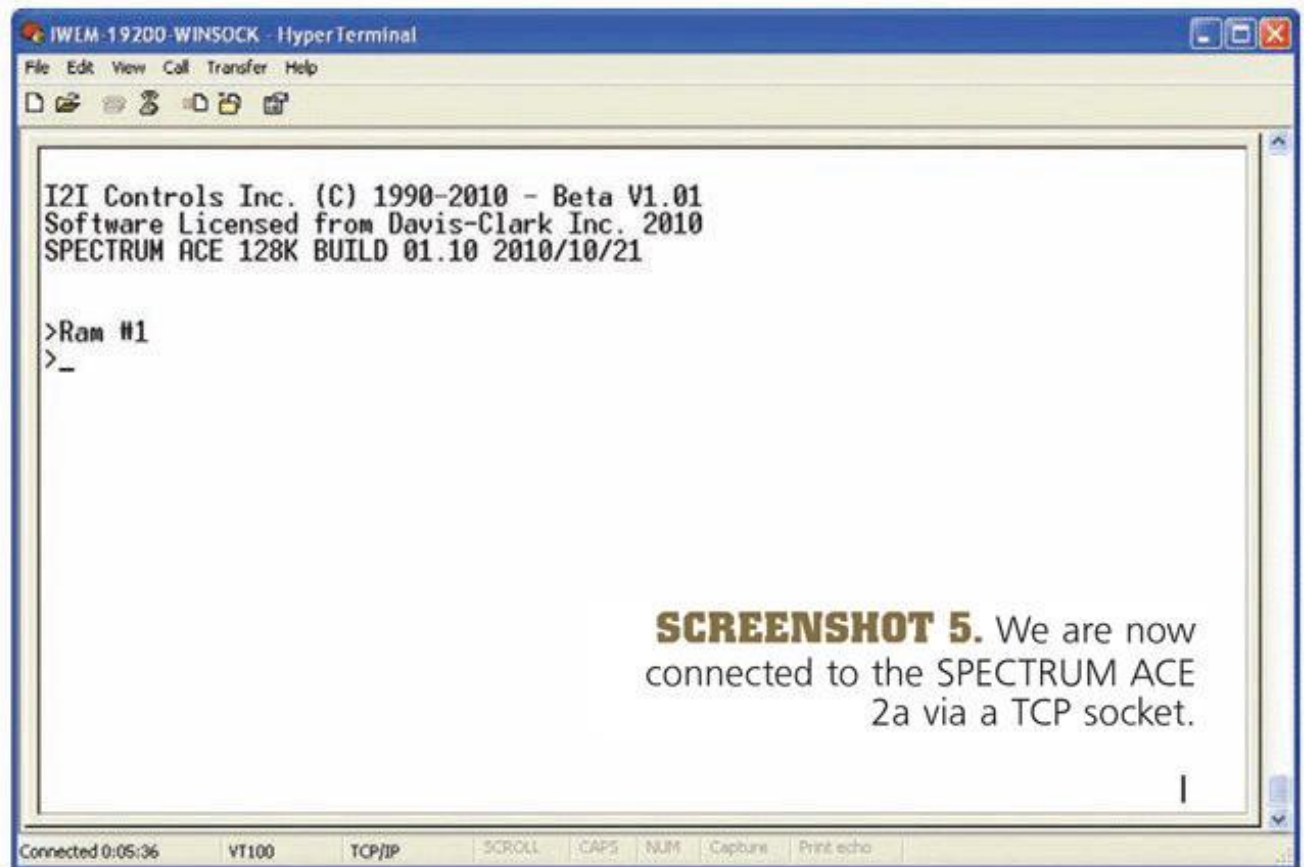


PHOTO 4. The Li-Ion battery pack I've chosen to use is powerful enough to support the iWEM-1001 for a long time at full power. However, the iWEM-1001 is designed to sleep and conserve battery life. So, a less juicy battery configuration can be substituted.



SCREENSHOT 4. The VT100 is one of the greatest terminals of all time. Everybody that is anybody in the terminal emulator business emulates this terminal or one of its many variants.



SCREENSHOT 5. We are now connected to the SPECTRUM ACE 2a via a TCP socket.

rate is the result of taking the default values in the WLAN data rate configuration.

The 802.11 up indicator is supported by the iWEM-1001's DHCP-assigned IP address of 192.168.0.112. The DHCP process also provided the network mask value of 255.255.255.0. The IP address of the GW (gateway) is actually the IP address of the router. The GW IP address is also provided courtesy of DHCP.

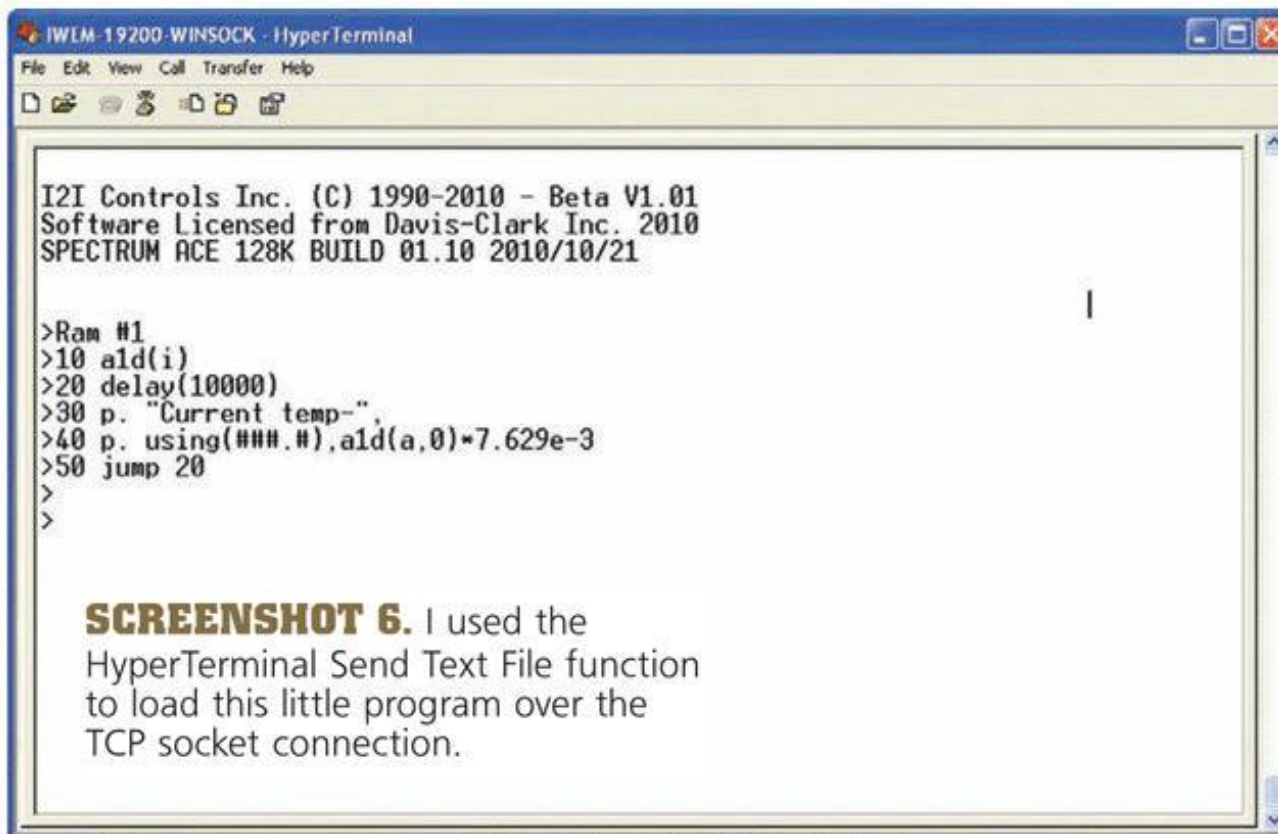
At this point, there are no external applications calling themselves host. The iWEM-1001 only knows about one host, which happens to be the router. The local and remote ports are at their default values and the iWEM-1001 is running its built-in TCP server application. As soon as I enter "exit," DATA will replace CMD and the iWEM-1001 will become a serial to ethernet bridge.

System Integration

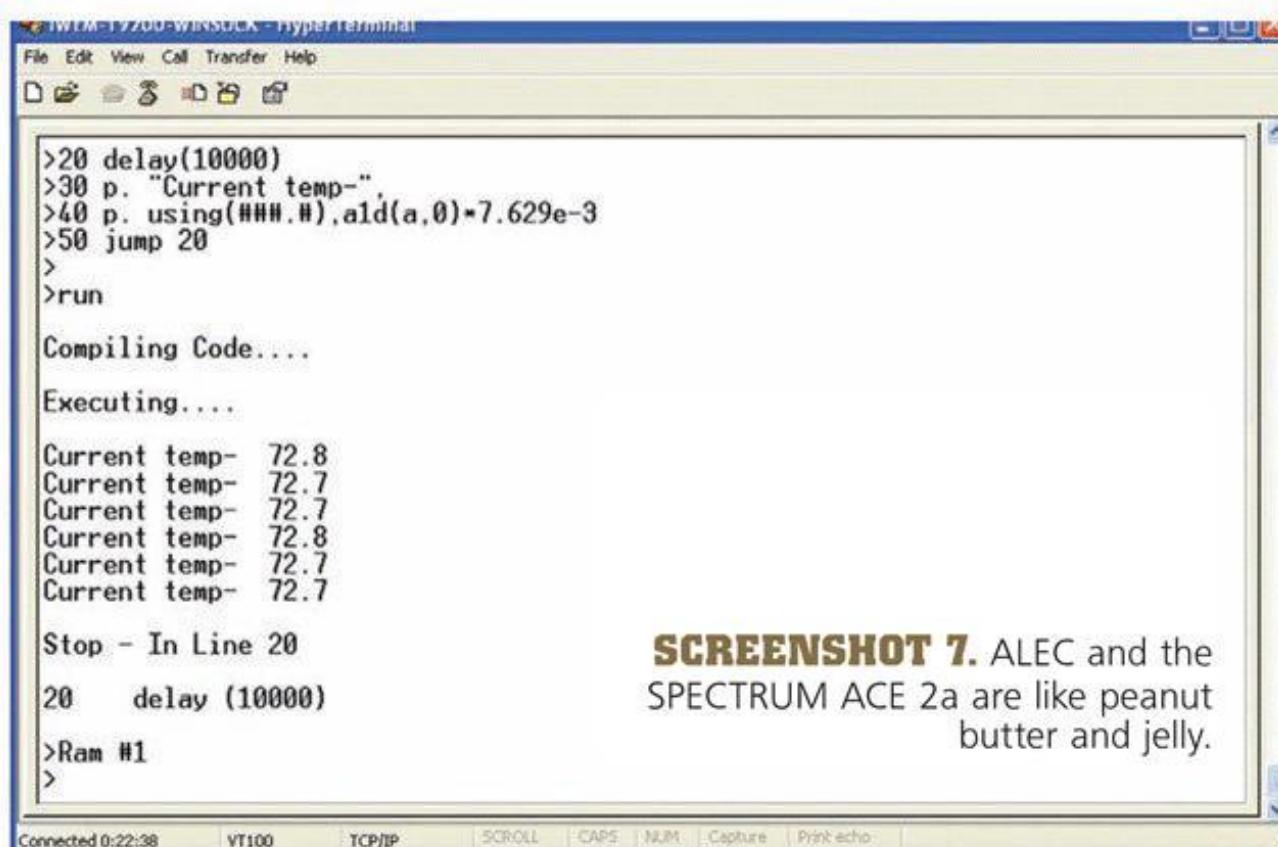
The point behind this discussion is to allow the SPECTRUM ACE 2a console port to be accessed from anywhere. The iWEM-1001 hookup is simple and easy. As you can see in **Photo 4**, the iWEM-1001 is powered by a 3.7 volt 6,600 mAh Li-Ion battery pack. The serial connection between the ACE 2a and iWEM-1001 is a no-brainer TX-RX-GND setup.

The ACE 2a works best with a 19200 HyperTerminal VT100 session. So, let's configure a HyperTerminal WINSOCK emulator session beginning with **Screenshot 2**. Instead of the normal serial interface, we're going to configure HyperTerminal to use an Ethernet interface.

We know that the iWEM-1001 was leased an IP address of 192.168.0.112 by the router on the EDTP network. We also know that the iWEM-1001's local port is 1000. The combination of its IP address and local port form a TCP socket. A TCP socket is not more than a communication endpoint. In this case, our TCP socket is a server socket that listens on port 1000 for incoming



SCREENSHOT 6. I used the HyperTerminal Send Text File function to load this little program over the TCP socket connection.



SCREENSHOT 7. ALEC and the SPECTRUM ACE 2a are like peanut butter and jelly.

requests from clients. We define our TCP socket to HyperTerminal as shown in **Screenshot 3**.

At this point, we can connect to the iWEM-1001. Once connected, we must select Properties from the File drop-down menu and select the Settings tab as shown in **Screenshot 4**. This is where we define the terminal emulation type. The ACE 2a wants to see a VT100 and that's what we'll appear to be. A tap of the spacebar results in **Screenshot 5** which is the ACE 2a banner. We have successfully contacted the ACE 2a via a TCP socket.

System Test

I wrote a small ALEC test program and loaded it in **Screenshot 6**. The ALEC code sets up the ACE 2a's A-to-D subsystem in line 10. A delay is honored in line 20. Lines 30 and 40 push the current temperature to the ACE 2a's console. The temperature data originates from an LM34. Upon entering "run" into the HyperTerminal TCP socket session, ALEC compiles and executes the temperature test program.

Screenshot 7 is the result of contacting the ACE 2a console, loading a program, executing a program, and viewing the results via a wireless TCP socket connection.

The Possibilities

Besides telling me why it's chilly in here, this little demonstration can also be used to tell me why it's chilly in Idaho. TCP sockets work on the Internet just like they do in a lab network contained within the confines of a single room. All you have to do to put the ACE 2a on the Internet is use a routable address and designate a local port address. The 192.xxx.xxx.xxx

Fred Eady can be reached via email at fred@edtp.com.

Sources

iWEM-1001
Lemos International
www.lemosint.com

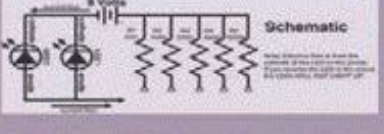
SPECTRUM ACE 2a
I2I Controls
www.i2icontrols.com

Li-Ion Battery Pack
Tenenergy Corporation
www.tenenergybattery.com

IP address range is not routable on the Internet. However, you can use the 192.xxx.xxx.xxx address range in your network behind the router. Most of today's routers employ NAT (Network Address Translation) functionality which allows you to literally punch a hole in the router to allow outside access to devices on the router's 192.xxx.xxx.xxx network. For instance, to put our iWEM-1001 configuration on the Internet, we would use the router's NAT functionality to open an Internet access portal to TCP socket 192.168.0.112:1000.

Whether you want to cut the wires locally via a LAN or internationally via the Internet, the iWEM-1001 is a foolproof and inexpensive way to go. **SV**

**FUNDAMENTALS** For The Beginner
Need the Basics?
Follow along with our series of articles which includes easy to understand graphics. Starting in the May 2010 issue.
Subscribe Today! (with optional digital back issue viewing.)
Visit www.nutsvolts.com or call (800) 783-4624



THE ORIGINAL SINCE 1994

PCB-POOL®

Beta LAYOUT

Servicing your complete PCB prototype needs :

- **Low Cost - High Quality** PCB Prototypes
- **Easy online Ordering**
- **Full DRC included**
- **Lead-times from 24 hrs**
- **Optional Chemical Tin finish** no extra cost

NEW !

NEW !

FREE LASER STENCIL WITH ALL PROTOTYPE PCB ORDERS

Watch "ur" PCB®
Follow the production of your PCB in **REALTIME**

email : sales@pcb-pool.com
Toll Free USA : 1 877 390 8541
www.pcb-pool.com

2006 **DESIGNER**

Adams **Graphic** **PROTEUS** **NATIONAL INSTRUMENTS**

Easy-PC **Spring Layout** **FOURD** **Incisive**

Beta LAYOUT

Programming The LEGO NXT:

An Alternative Approach Suitable For Developing Tomorrow's Engineers

by John Blankenship and Samuel Mishal

No one makes building a robot easier than LEGO, and now, *programming* the LEGO NXT can be just as easy. The open-source LegoLibrary.bas described here makes it easy to introduce programming concepts to those new to robotics. Including the library in your RobotBASIC programs enables them to directly control the robot without downloading any programs to the NXT. This easy-to-use system creates an alternative motivational environment ideal for cultivating tomorrow's engineers.

www.servomagazine.com/index.php?/magazine/article/june2011_Blankenship

One of the most important things robot enthusiasts can do is share their love of robotics with young hobbyists who may be destined to create the next generation of intelligent machines. Instilling aspirations of

building a robot though, is simply not enough. Learning to program a robot should be the ultimate goal for students because programming promotes logical thinking and develops problem-solving skills. What is needed is a robot that is easy to build and easy to program.

As mentioned, no one makes building a robot easier than LEGO. With their system, anyone can snap together a robot base and power it with modular motors. A variety of sensors can be interfaced by simply connecting them to the LEGO processor with a cable. The visual programming language that ships with the NXT robot was selected to make programming easy, but many users find procedural languages more intuitive.

One advantage of a visual programming environment is that the programmer does not have to deal with the complexities of implementing flow control structures such as loops and decision statements. This can make visual programming very efficient for experienced programmers. Unfortunately, removing the complexities can also prevent some beginning programmers from truly understanding the logic of what is

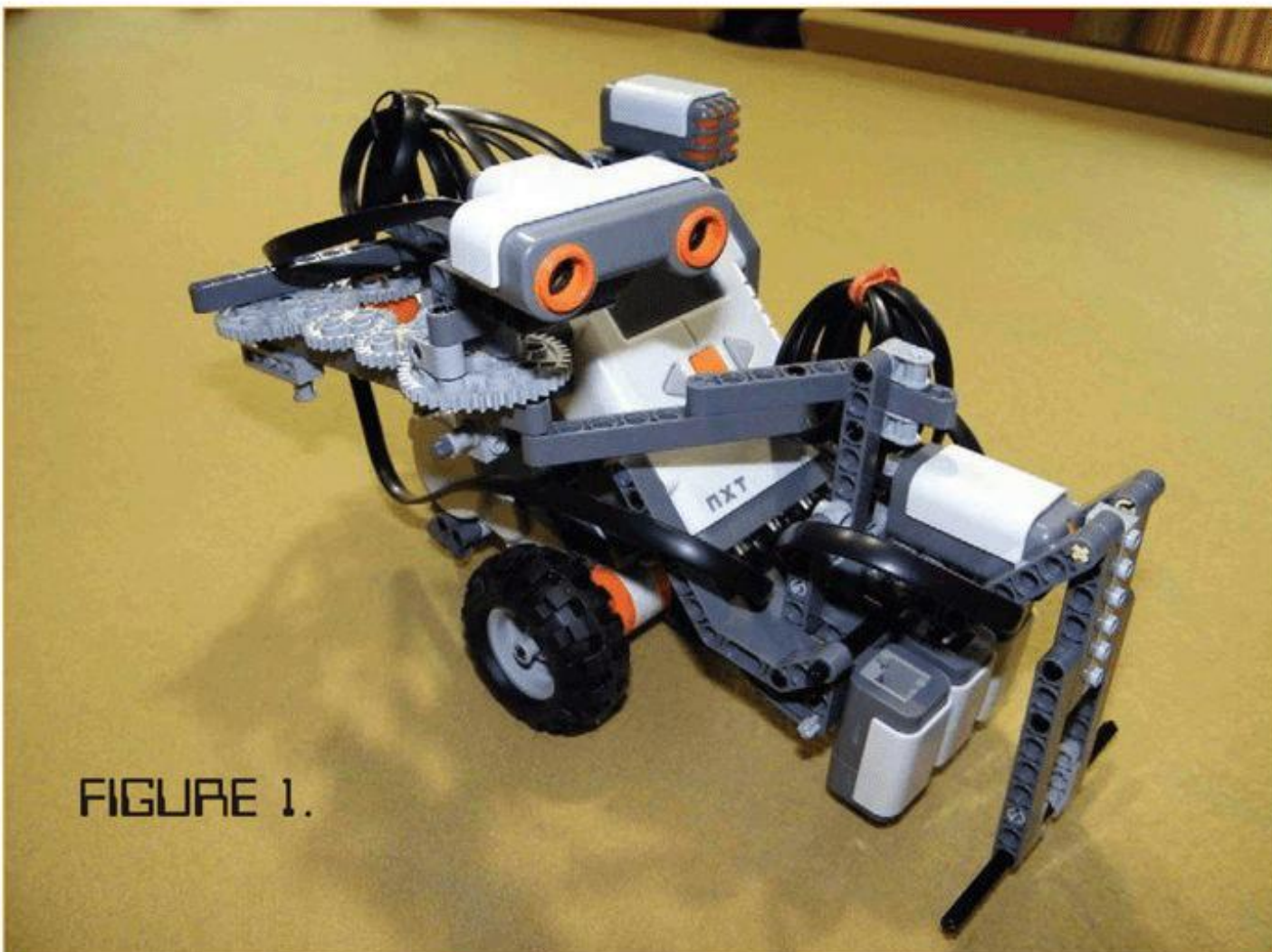


FIGURE 1.

happening. If you want proof of this assertion, watch children learn to program with a visual language. They generally can create programs quickly by simply duplicating pictures, but ask them to modify the program to create an alternative behavior and watch their reaction.

It is important to stress that we are not saying that visual languages are without value. For many situations, they provide an ideal solution and visual learners may learn better when using them.

Procedural languages, though, make the flow of the program much easier to grasp for many conventional users. Regrettably, though, procedural languages generally force programmers to immerse themselves in many other details in order to accomplish even simple tasks. In order to remedy this problem, we created a library of routines that hide the details of controlling LEGO's motors and reading their sensors.

The library routines are generic enough that they can be used with a variety of configurations of the NXT robot. **Figure 1** shows the robot we used to develop and test the routines. The LEGO computer has built-in Bluetooth hardware and a communication protocol that our routines use to control the robot directly without downloading any programs to the robot itself. This feature alone makes it far easier for many beginners to program the LEGO robot. (See this month's *The NXT Big Thing* column.)

Let's look at a simple program that demonstrates how the library can make programming an NXT easy, but more importantly, it demonstrates how programs written with the library are easily understood by beginners. The program in **Figure 2** is written in RobotBASIC — a free language available from www.RobotBASIC.com.

The first line in the program includes the LegoLibrary, so that our routines become a part of the program. The second line initializes a variable to the number of the Bluetooth port used by our machine. The third line initializes the library and brings us to the main body of the program.

A call to the library routine **LegoDriveMotors** turns both motors on at a **FAST** speed. The first parameter controls the left wheel and the second controls the right. The next line calls another library routine to produce a 3,000 ms pause before the motors are turned off. Nearly anyone can quickly understand how this program works, and once told they can use a speed of **SLOW** as well as **FAST**, they can easily make the robot move at different speeds for different periods of time. They can also make the robot turn by stopping or slowing down one wheel while the other moves at a **FAST** pace. This open-loop control allows novice hobbyists or even children to program a robot to move in various patterns. We have successfully taught fifth graders the fundamentals of programming using this approach.

After a little practice with open-loop control, it is easy to use LEGO's sensors to control the robot's behavior. The program in **Figure 3**, for example,

```
#include "LegoLibrary.bas"
BluetoothPort = 34
call LegoInit(BluetoothPort)
call LegoDriveMotors(FAST,FAST)
call Wait(3000)
call LegoDriveMotors(STOP,STOP)
end
```

FIGURE 2.

causes the robot to move forward until the ultrasonic ranging sensor detects an object less than five inches away. When this happens, the robot turns a random amount and continues the behavior.

Our LegoLibrary contains routines for interrogating LEGO's line sensor, sound sensor, bumper sensor, as well as the ranging sensor. The routines must be past the port number where the sensor is connected. This allows the utilization of multiple sensors if you choose. For example, you could create a robot with four ranging sensors.

The source code for the library can be downloaded from www.RobotBASIC.com and anyone with programming experience can easily identify all the available routines by looking through the listing. Those new to programming might benefit from the book *RobotBASIC Projects for the LEGO NXT*. It uses the library routines to develop robot behaviors for avoiding objects, following lines, hugging walls, and more.

A LegoSimulationLibrary is also available from the web page. When it is included instead of the standard library, the same programs control a simulated robot allowing experimentation even when hardware is not available. This can be especially valuable for schools because every student can work with their own simulated robot. When their program is working, the teacher can plug in a Bluetooth adapter to allow control of the real robot.

If you are looking for a more conventional language to control your LEGO NXT or if you just want to share your enthusiasm with someone totally new to robotics, give our LegoLibrary a try. **SV**

```
#include "LegoLibrary.bas"
BluetoothPort = 34
RangePort = 3
call LegoInit(BluetoothPort)
call LegoRangeInit(RangePort)
while true
  call LegoRangeSensor(RangePort,r)
  // Places the distance measured into the variable r
  if r < 5
    call LegoDriveMotors(SLOW, -SLOW) // rotate robot
    call Wait(1500 + random(800)) // for a random time
  else
    call LegoDriveMotors(SLOW, SLOW)
  endif
wend
end
```

FIGURE 3.



The NXT Big Thing #11

In Control!

By Greg Intermaggio



Hey, hey, hey,
everybody! Welcome to
this edition of The NXT
Big Thing that'll end with
you using a remote to
control Eddie! This
month, we'll be learning
how to make two NXTs
communicate over
Bluetooth, so that one
can control the other.



Get ready to get busy, as we learn just how much programming has to go into both the remote control and the robot itself to make it go.

NXT Prep

We'll need two things before we get started programming:

- A fully-built Eddie 2.0; you can find Eddie 2.0 building instructions in the January '11 edition.
- A second, plain NXT with a touch sensor attached. Plug it into port 1 and attach it to the NXT any way you like.

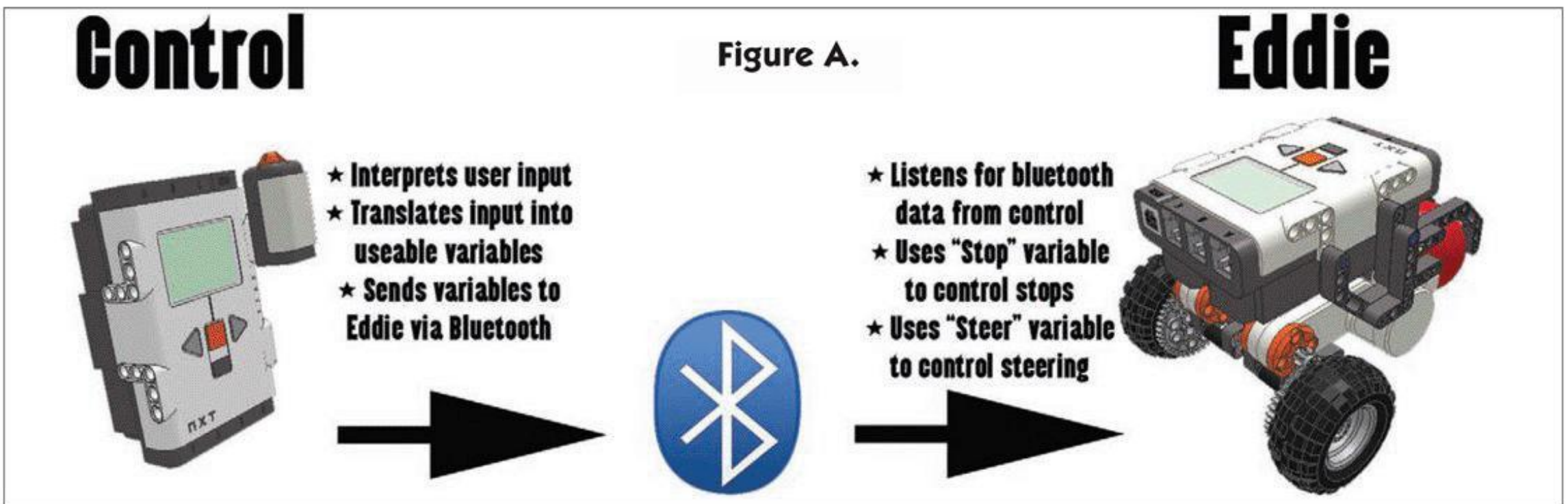
Understanding The Program

As I said before, there are two components going into this project.

The first component is the remote control. It takes your input from pressing buttons, translates it into something Eddie can read, and sends it to him over Bluetooth.

The second component is Eddie. Eddie listens for the messages from the controller, and uses the variables sent to it to determine whether or not to move, and which direction to move in.

Figure A shows the flow of information in a bit more detail.



Writing The Control Program

Now that we understand how the programs work, let's start by writing the program for the remote control.

Control Test Program Instructions

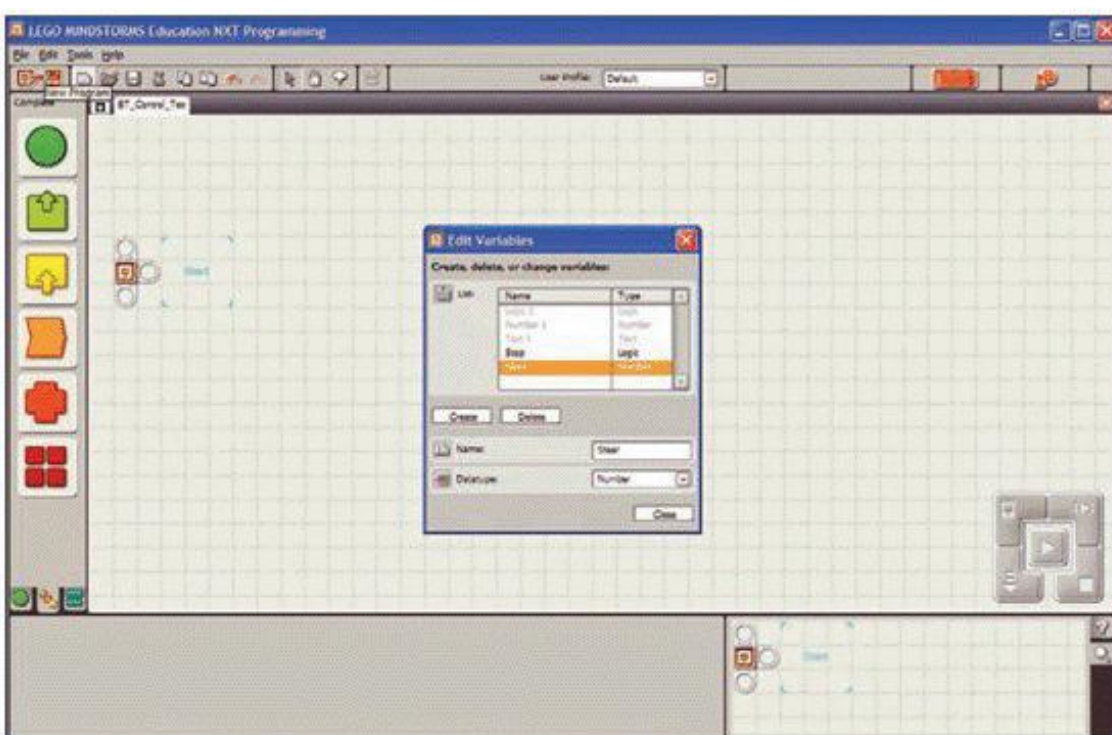


Figure 1. Start by creating a new program called BT_Control_Test. Then, click Edit > Manage Variables. Add a logic variable called STOP, and a number variable called Steer. Stop will control whether Eddie moves or stops. Steer will control which direction Eddie moves.

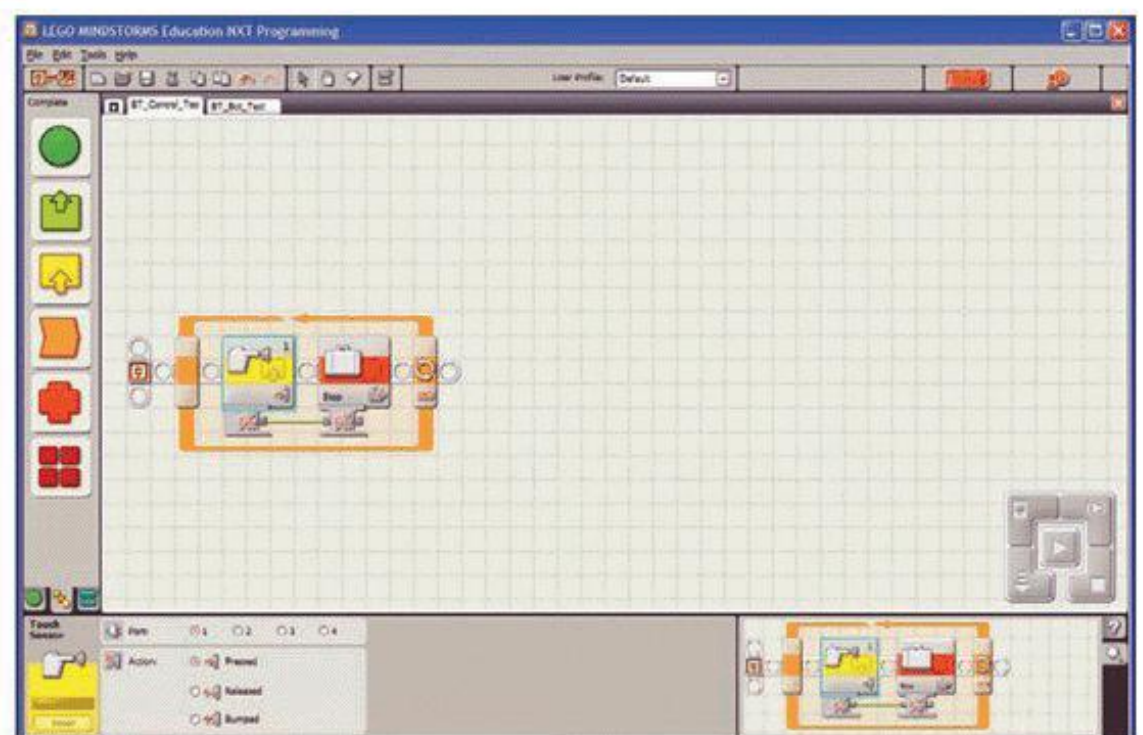


Figure 2. Add a loop. Inside the loop, drag a touch sensor block in. Add a variable block and choose the Stop variable. Set the variable to write instead of read. Finally, wire the output of the touch sensor block to the input of the variable block. This means that when the touch sensor is pressed, Stop will be set to true.

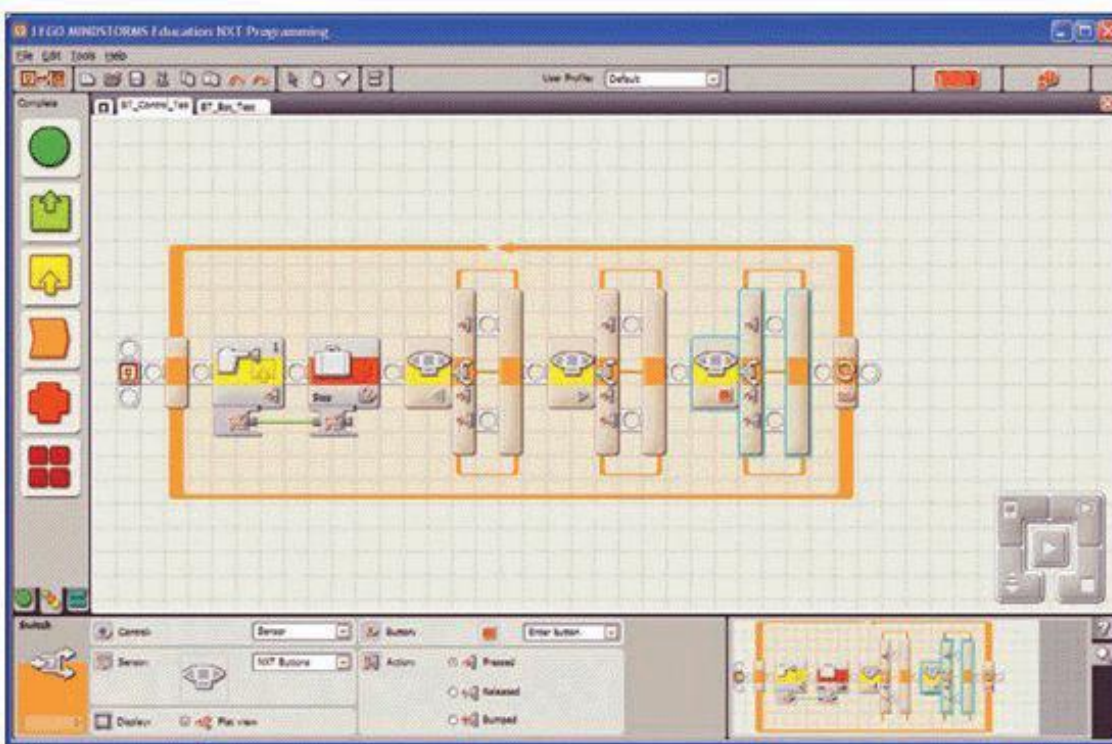


Figure 3. Add three switches set to "NXT Buttons" for the sensor. Set the first switch to the left arrow; the second to the right arrow; and the third to the Enter button.

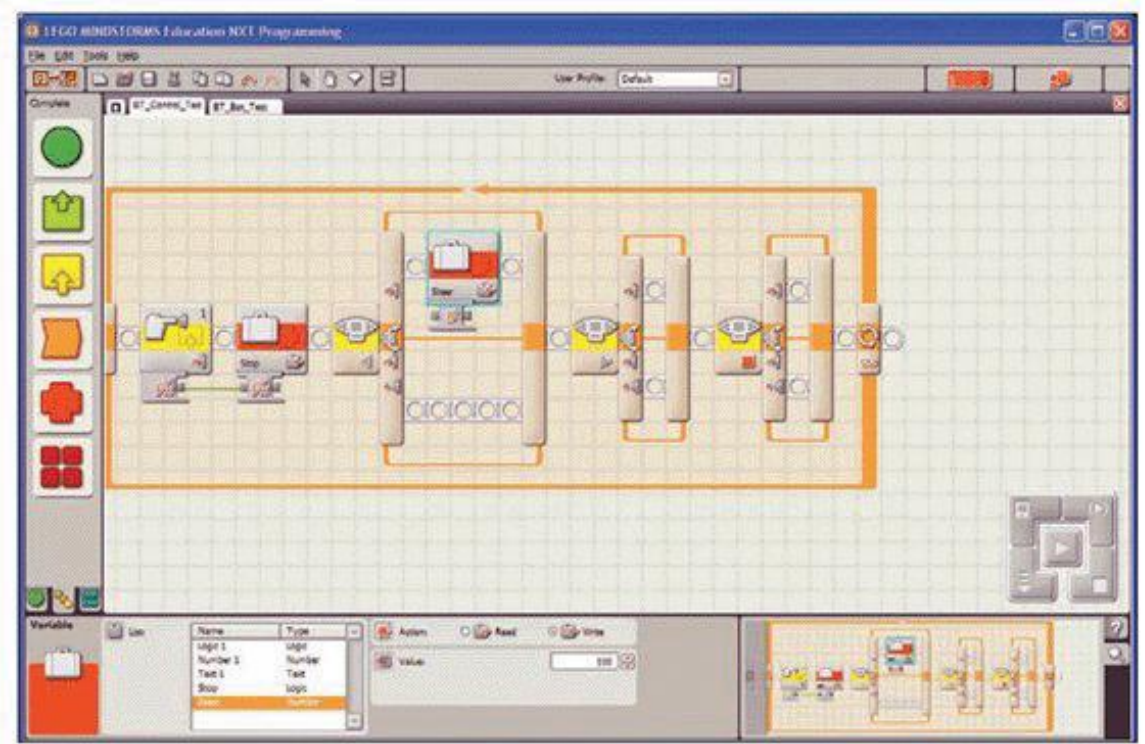


Figure 4. Add a variable block to the side of the left arrow switch where the button is pressed. Set that block to the variable Steer and set it to write. Finally, set the value to 100. Recall that "steering" can be controlled by an integer between -100 and 100; -100 is a full right turn (it's reversed from a full left turn since Eddie has a gear train); 100 is a full left turn. When the left arrow is pressed, we'll ultimately want Eddie to turn left.

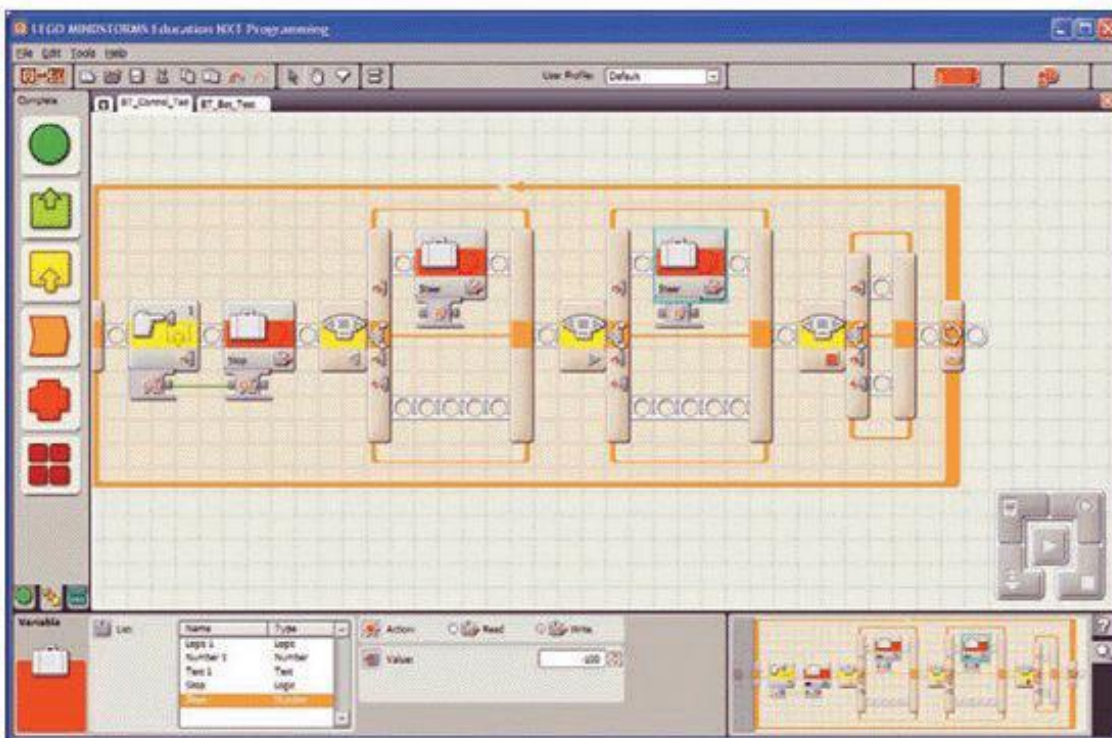


Figure 5. Add the same Steer variable block to the right NXT button switch. This time, set it to -100 for a full right turn.

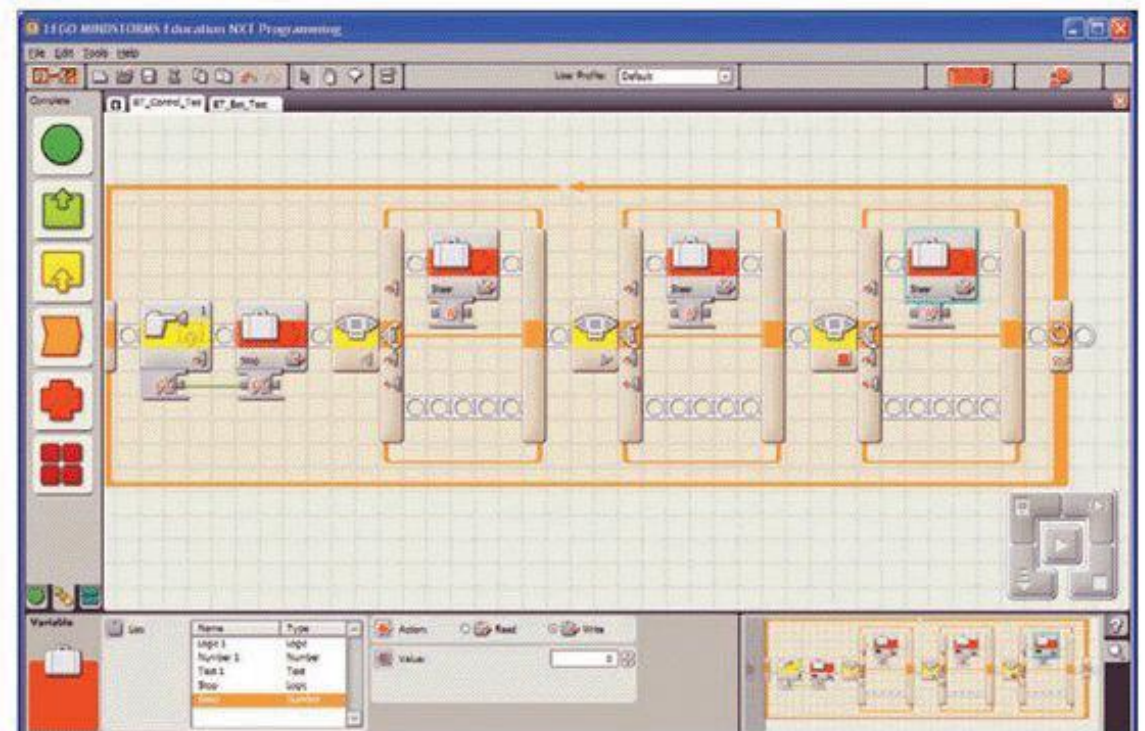


Figure 6. Again, add the Steer variable block to the Enter button switch. This time, set it to 0 to make Eddie go straight forward when the Enter button is pressed.

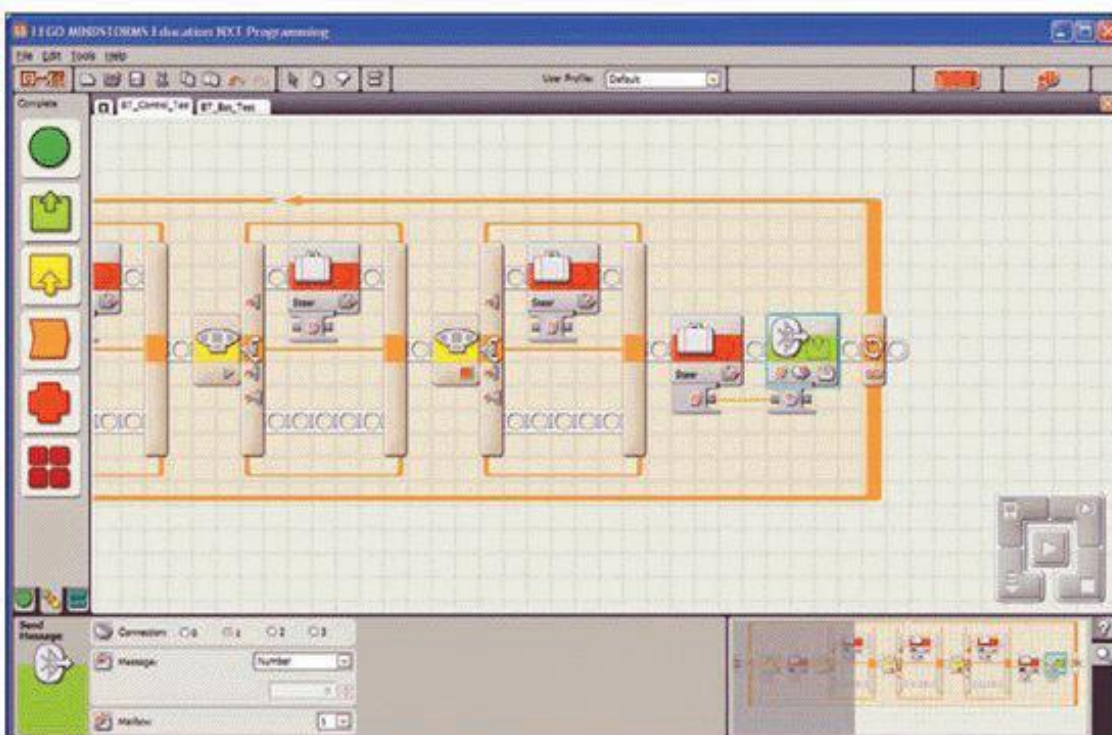


Figure 7. Add a variable block to the end of the program and set it to the Steer variable. Next, find and add a Send Message block to the end of the program from the Actions menu. Set the connection number to 1, the message to logic, and the mailbox to 1. Wire the value from the variable block to the number data hub on the Send Message block. You'll have to connect the two NXTs via Bluetooth manually, at which point you choose the connection number. The mailbox number is where the data is sent, and we'll use it later to tell Eddie where to look for those important variables.

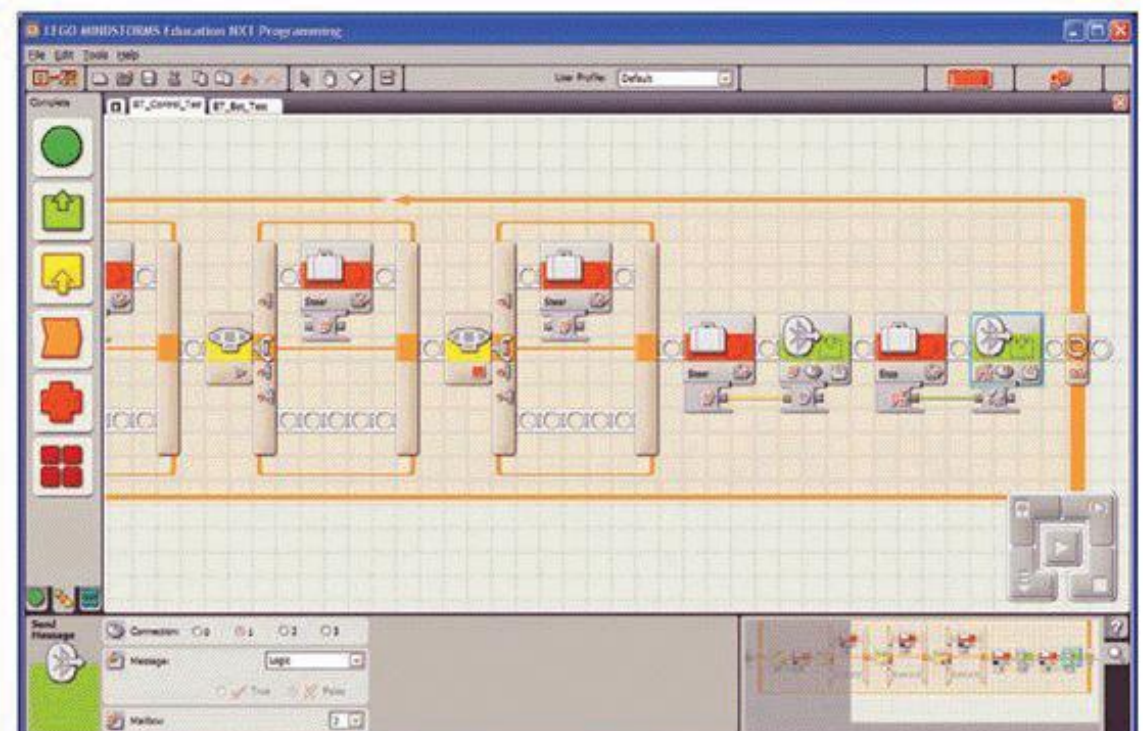


Figure 8. Add a variable block and set it to the Stop variable. Add another Send Message block, and this time set it to connection port 1, message "Number," and mailbox 2. Then, connect the value from the variable block to the logic data hub on the Send Message block.

Writing Eddie's Program

We're halfway there! Now, we just need to program Eddie to understand the Bluetooth messages.

Bot Test Program Instructions

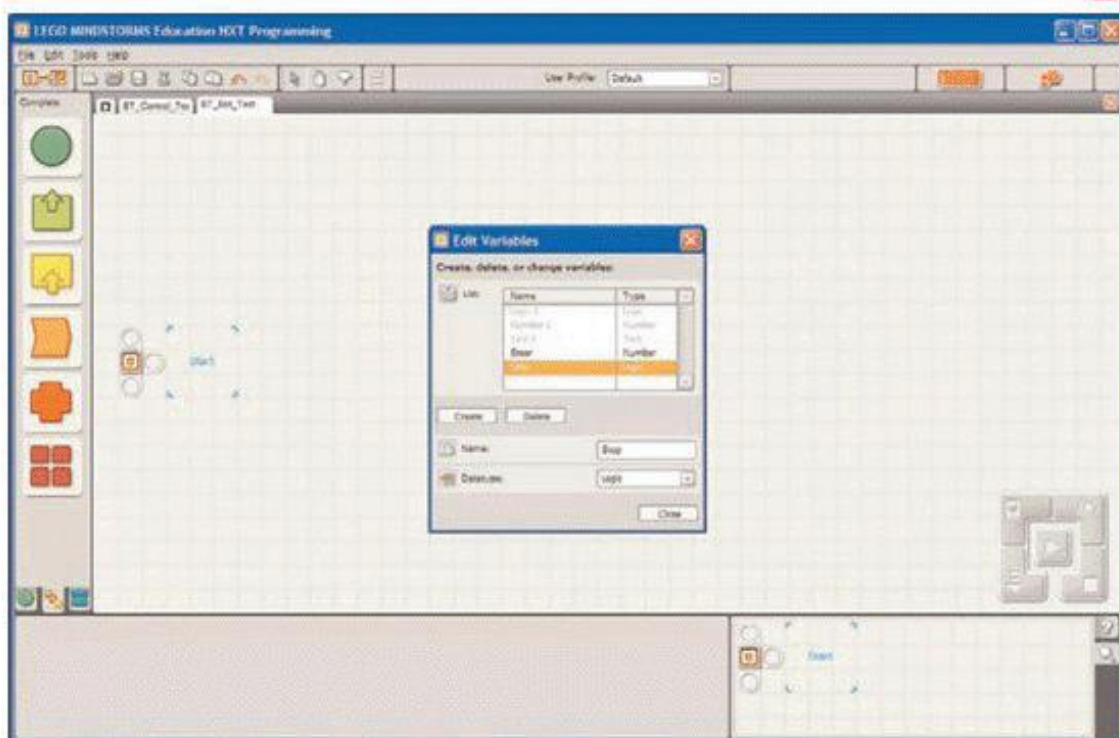


Figure 1. Create a new program called BT_Bot_Test. Define a number variable called Steer and a logic variable called Stop – these are the same variables from the Control program.

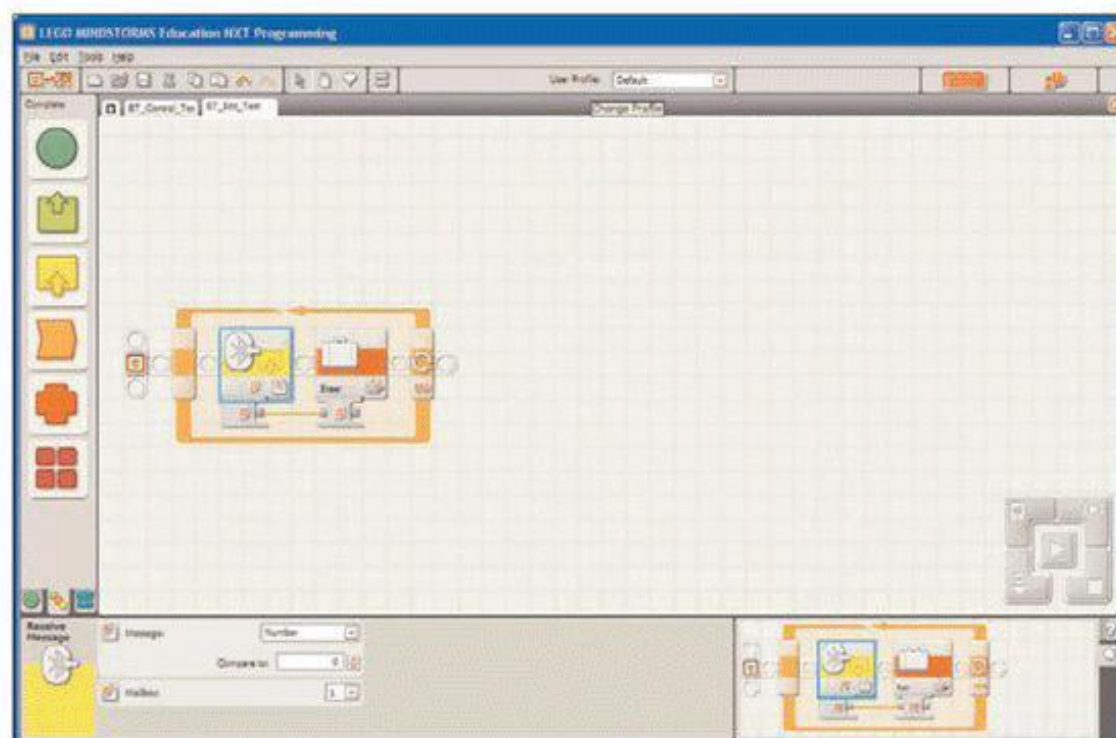


Figure 2. Add a loop. Inside the loop, add a Receive Message block from the Sensors tab. Set the mailbox to 1 and set it to the Number Out data hub to write to a Steer variable block.

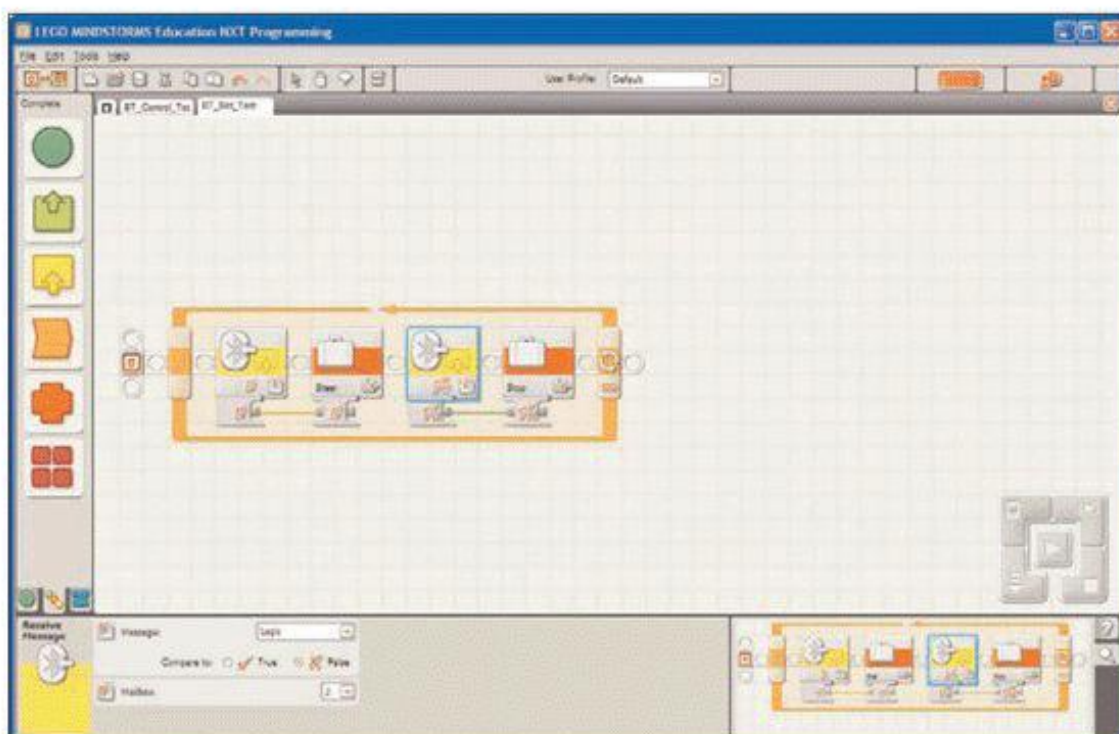


Figure 3. Repeat the same process as Step 2, but set the mailbox to 2 and the variable to Stop. Wire the Logic Out data hub to the Stop variable.

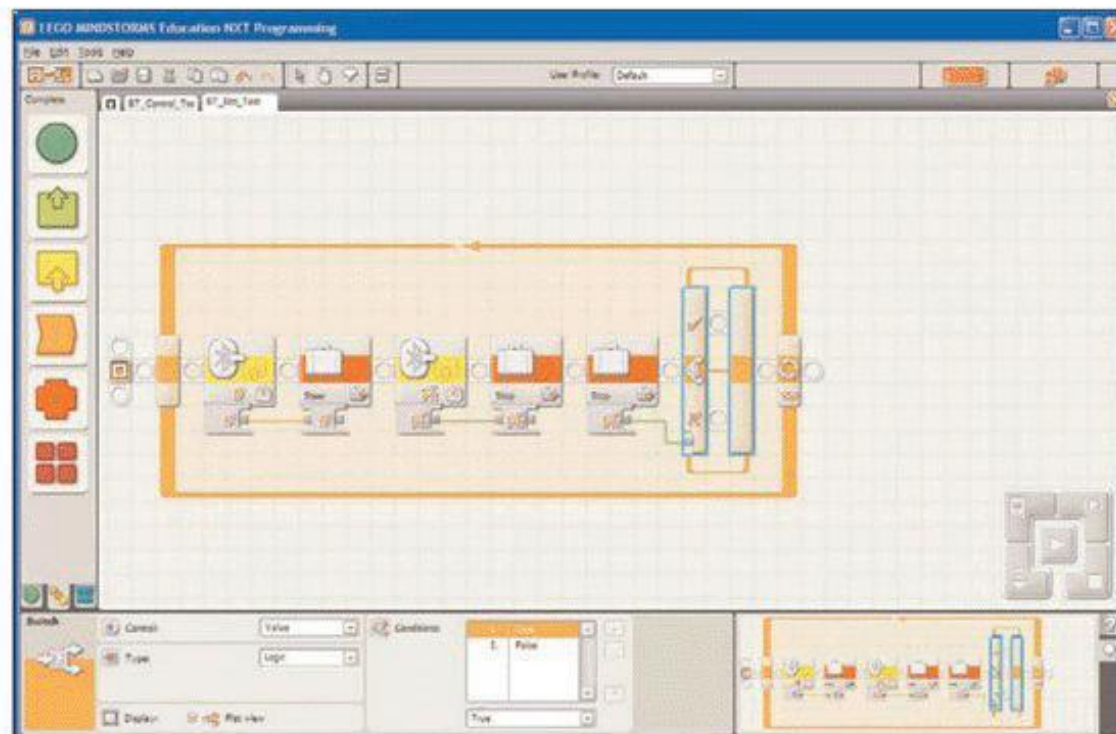


Figure 4. Add a Stop variable block set to read and wire it to a switch set to logic. This will allow us to make Eddie react differently if the touch sensor is being pressed (meaning "Stop" is set to "true") than if the touch sensor isn't being pressed (meaning "Stop" is set to "false").

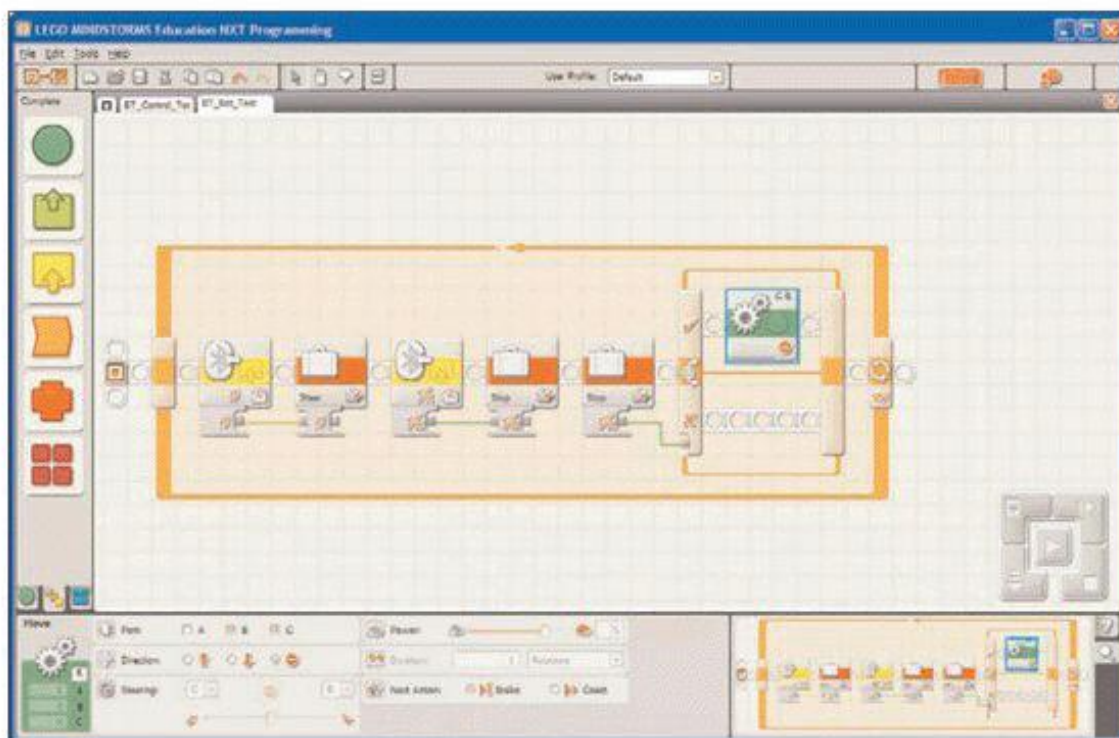


Figure 5. If "Stop" is set to true (meaning the touch sensor is being pressed), we want Eddie to stop moving. Add a Move block set to Stop on the true side of the logic switch.

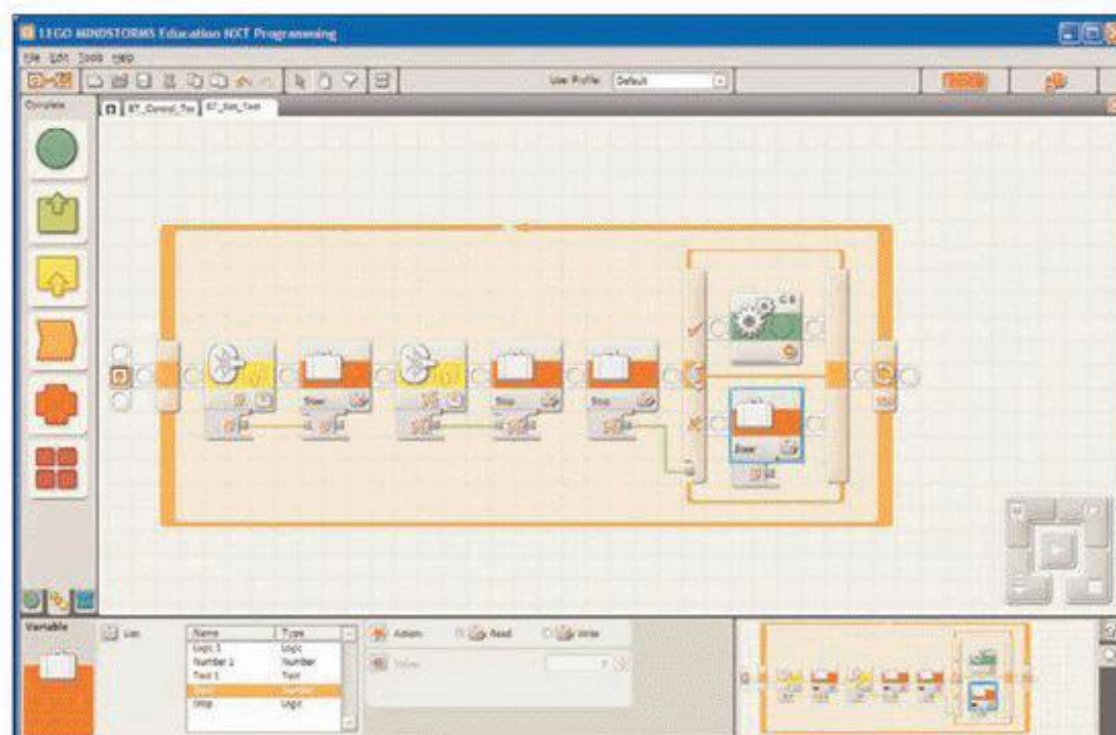


Figure 6. Add a Steer variable block to the false side of the logic switch.

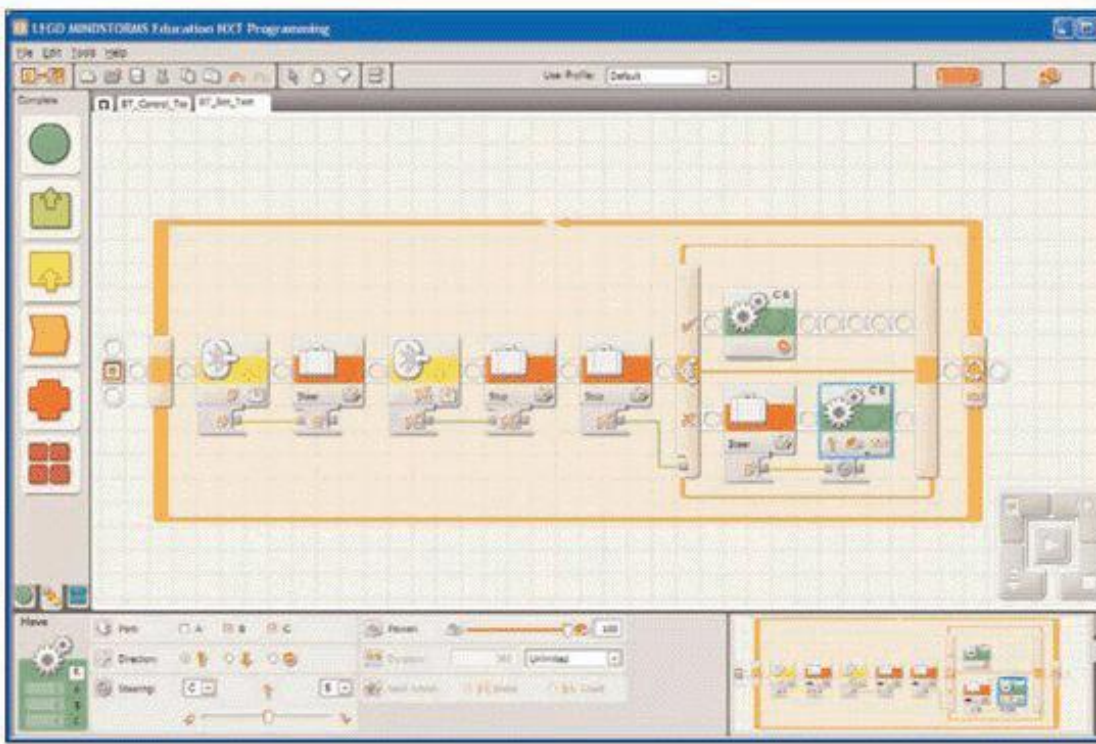


Figure 7. Add a Move block, and wire the output of the Steer variable to the Steering data hub of the Move block.

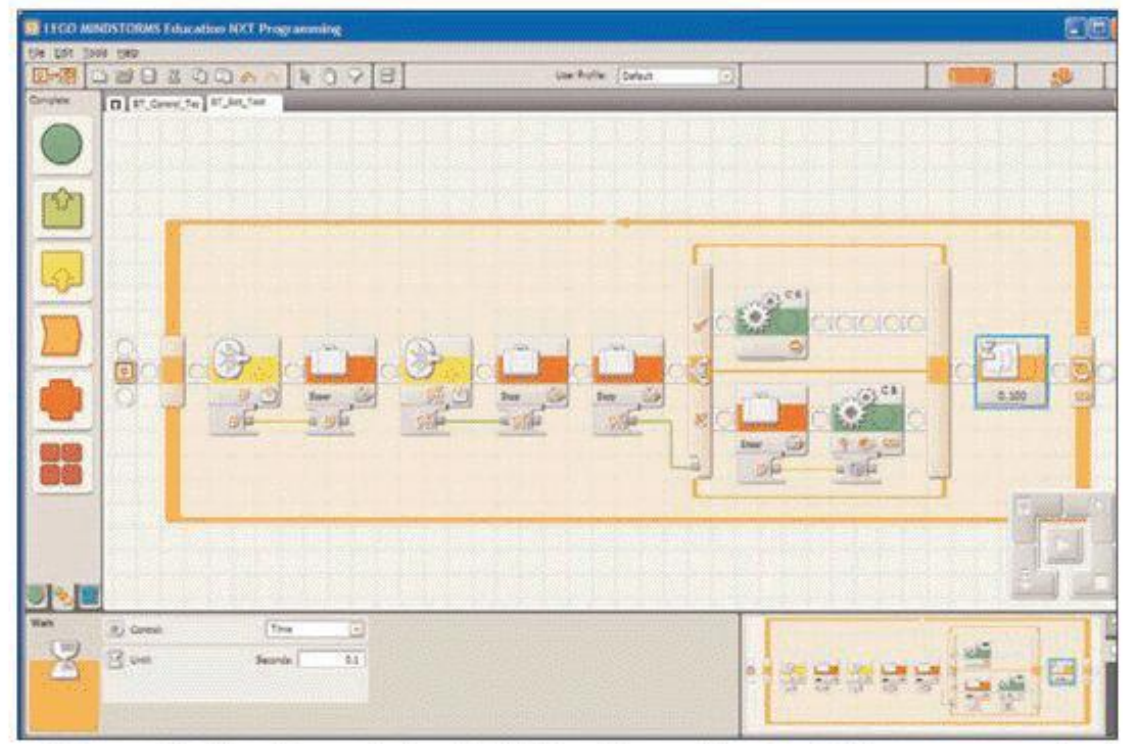


Figure 8. Finally, add a Wait For Time block at the very end of the program, setting it for .1 seconds. This will make sure that Eddie doesn't try to look for Bluetooth signals too fast, when they aren't there yet.

Finishing Touches

So close! It's time to download our programs and configure our Bluetooth. Here's how:

- Download BT_Control_Test to the controller.
- Download BT_Bot_Test to Eddie.
- Access the main menus on both NXTs.
- Go to Bluetooth and make sure both NXTs have Bluetooth both On and Visible.

- On one NXT, select Search from the Bluetooth menu. When the other NXT shows up in the search results, select it and connect.
- Now both devices are connected. If either one is turned off, you'll have to re-establish the connection by going into Bluetooth > My Contacts, and selecting the other NXT.
- Run the control program on the remote control, and the bot program on Eddie.

If all goes well, pressing the orange Enter button on the controller should make Eddie move forward; either arrow should make him turn in the specified direction; and holding the touch sensor should make him stop in place until it's released.

Wrapping Up

We just learned how to connect two NXTs via Bluetooth, and make one act as a remote control for the other. In the next edition of The NXT Big Thing, we'll take Bluetooth control even further — so stay tuned!

In the meantime, here's a few cool challenge ideas for you to keep your brain in gear:

- Make Eddie's remote use a variety of sensors instead of the NXT buttons.
- Program Eddie to play a sound when a certain sensor/button is used.
- Attach a wireless camera, and have Eddie bring in your newspaper. **SV**



HiTechnic

The only complete range
of LEGO Certified sensors and
accessories for LEGO MINDSTORMS



NXT Color Sensor V2



NXT Angle Sensor

For a complete list
of all our products
please visit our
website.

Email us: sales@hitechnic.com

www.HiTechnic.com



Greg "LEGO" Intermaggio lives in the Bay Area, CA, where he runs a business called Techsplosion, bringing hands-on science to all ages and walks of life. In his spare time, Greg likes to unicycle, juggle, unicycle juggle, and battle killer robots! More information about Greg can be found at Intermaggio.com. More information about Techsplosion can be found at Techsplosion.org

The *SERVO* Webstore

Attention Subscribers ask about your discount on prices marked with an *

CD-ROM SPECIALS



Free Shipping!

7 CD-ROMs & Hat Special

Only \$ 149.95 or \$24.95 each.

www.servomagazine.com



2010 CD ROM

On Sale Now!

\$24.95

We'll ship for FREE!

ROBOTICS

PIC Robotics

by John Iovine

Here's everything the robotics hobbyist needs to harness the power of the PICMicro MCU!

In this heavily-illustrated resource, author John Iovine provides plans and complete parts lists for 11 easy-to-build robots each with a PICMicro "brain." The expertly written coverage of the PIC Basic Computer makes programming a snap – and lots of fun.

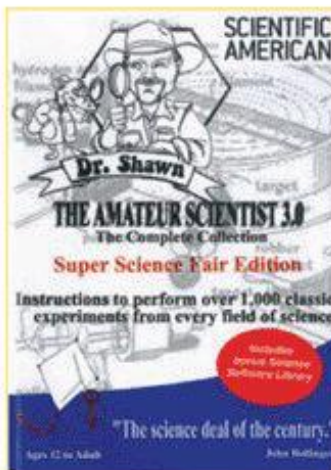
\$24.95

The Amateur Scientist 4.0 The Complete Collection

by Bright Science, LLC

There are 1,000 projects on this CD, not to mention the additional technical info and bonus features. It doesn't matter if you're a complete novice looking to do your first science fair project or a super tech-head gadget freak; there are enough projects on the single CD-ROM to keep you and 50 of your friends busy for a lifetime!

Reg \$26.95 Sale Price \$23.95

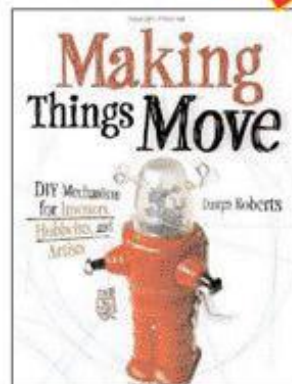


Making Things Move: DIY Mechanisms for Inventors, Hobbyists, and Artists

by Dustyn Roberts

In *Making Things Move: DIY Mechanisms for Inventors, Hobbyists, and Artists*, you'll learn how to successfully build moving mechanisms through non-technical explanations, examples, and do-it-yourself projects – from kinetic art installations to creative toys to energy-harvesting devices. Photographs, illustrations, screenshots, and images of 3D models are included for each project.

\$29.95*



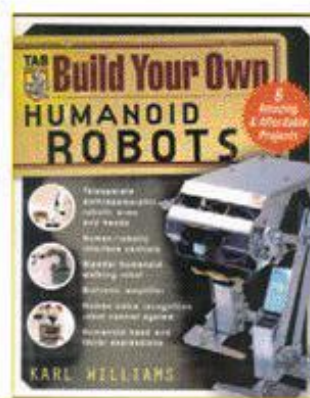
Build Your Own Humanoid Robots

by Karl Williams

GREAT 'DROIDS, INDEED!

This unique guide to sophisticated robotics projects brings humanoid robot construction home to the hobbyist. Written by a well-known figure in the robotics community, *Build Your Own Humanoid Robots* provides step-by-step directions for six exciting projects, each costing less than \$300. Together, they form the essential ingredients for making your own humanoid robot.

\$24.95*

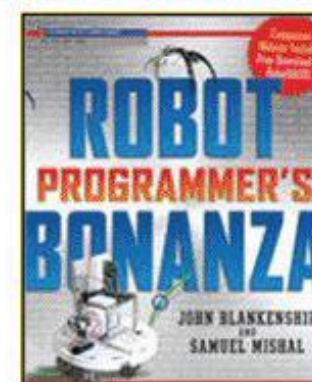


Robot Programmer's Bonanza

by John Blankenship, Samuel Mishal

The first hands-on programming guide for today's robot hobbyist! Get ready to reach into your programming toolbox and control a robot like never before! *Robot Programmer's Bonanza* is the one-stop guide for everyone from robot novices to advanced hobbyists who are ready to go beyond just building robots and start programming them to perform useful tasks.

\$29.95

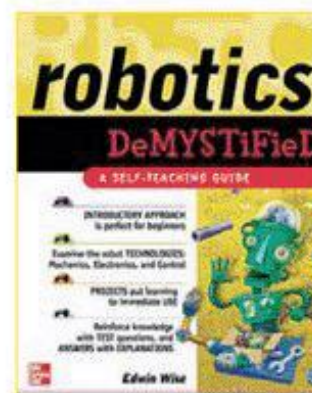


Robotics Demystified

by Edwin Wise

YOU DON'T NEED ARTIFICIAL INTELLIGENCE TO LEARN ROBOTICS! Now anyone with an interest in robotics can gain a deeper understanding – without formal training, unlimited time, or a genius IQ. In *Robotics Demystified*, expert robot builder and author Edwin Wise provides an effective and totally painless way to learn about the technologies used to build robots!

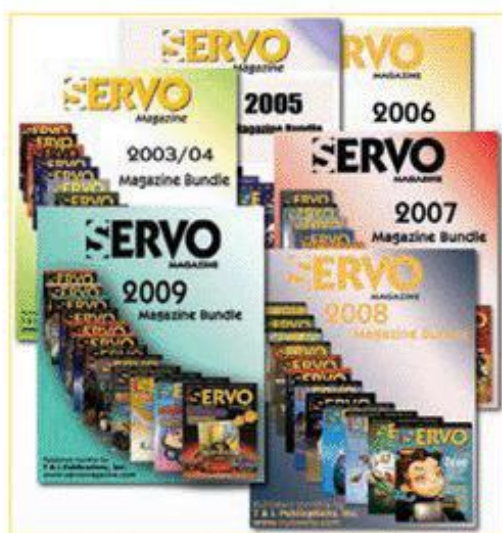
\$19.95



We accept VISA, MC, AMEX, and DISCOVER
Prices do not include shipping and may be subject to change.

To order call 1-800-783-4624

SERVO Magazine Bundles



Published by T & L Publications, Inc.

\$57
per bundle

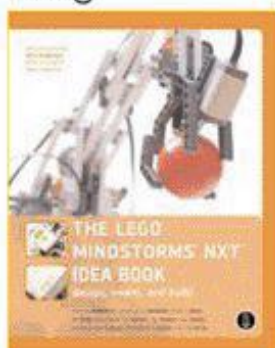
Save \$10
off the
normal
price!!

Now you can get one year's worth of all your favorite articles from *SERVO Magazine* in a convenient bundle of print copies. Available for years 04, 05, 06, 07, 08, and 09.

LEGO MINDSTORMS NXT Idea Book

by the Contributors to
The NXT Step Blog

If you're serious about having fun with LEGO® robotics, you've come to the right place. The team behind The NXT STEP blog — the authoritative online source for MINDSTORMS® NXT information and advice — has packaged its considerable skills and experience in this book. Inside, you'll find some of the team's best ideas for creating cool and sophisticated models, including instructions for eight robots you can build yourself.

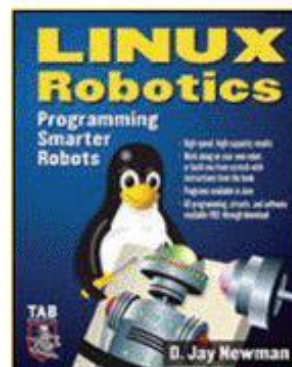


Reg \$29.95 Sale Price \$24.95

Linux Robotics

by D. Jay Newman

If you want your robot to have more brains than microcontrollers can deliver — if you want a truly intelligent, high-capability robot — everything you need is right here. *Linux Robotics* gives you step-by-step directions for "Zeppo," a super-smart, single-board-powered robot that can be built by any hobbyist. You also get complete instructions for incorporating Linux single boards into your own unique robotic designs. No programming experience is required. This book includes access to all the downloadable programs you need.

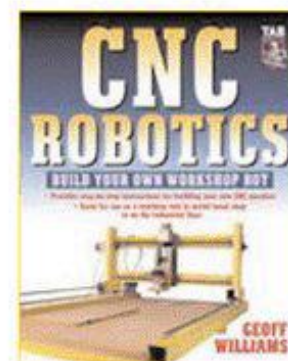


\$34.95

CNC Robotics

by Geoff Williams

Here's the FIRST book to offer step-by-step guidelines that walk the reader through the entire process of building a CNC (Computer Numerical Control) machine from start to finish. Using inexpensive, off-the-shelf parts, readers can build CNC machines with true industrial shop applications such as machining, routing, and cutting — at a fraction of what it would cost to purchase one. Great for anyone who wants to automate a task in their home shop or small business. **\$34.95**

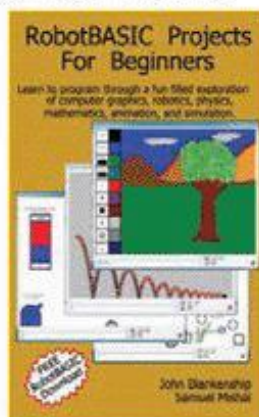


SPECIAL OFFERS

RobotBASIC Projects For Beginners

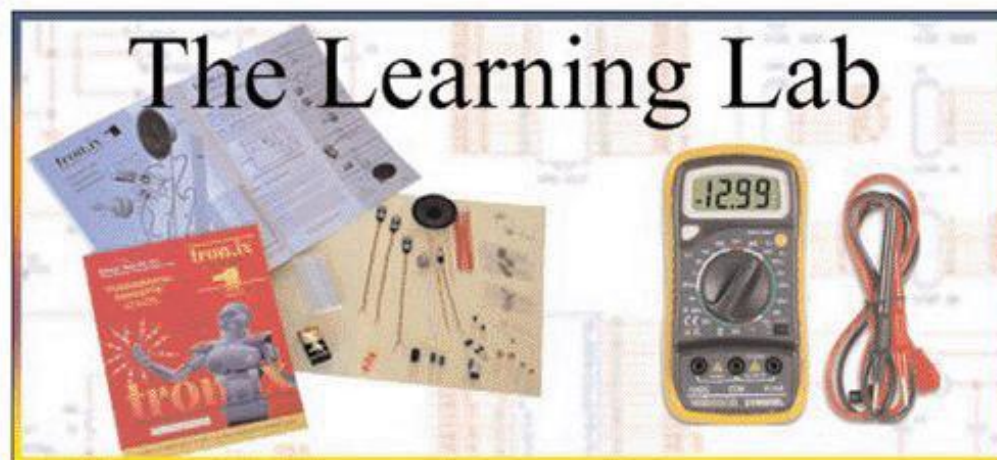
by John Blankenship, Samuel Mishal

If you want to learn how to program, this is the book for you. Most texts on programming offer dry, boring examples that are difficult to follow. In this book, a wide variety of interesting and relevant subjects are explored using a problem-solving methodology that develops logical thinking skills while making learning fun. RobotBASIC is an easy-to-use computer language available for any Windows-based PC and is used throughout the text.



Reg. Price \$14.95 Sale Price \$9.95

The Learning Lab



Technology Education Package for Everyone Starting in Electronics

This lab — from the good people at GSS Tech Ed — will show you 40 of the most simple and interesting experiments and lessons you have ever seen on a solderless circuit board. As you do each experiment, you learn how basic components work in a circuit. Along with the purchase of the lab, you will receive a special password to access the fantastic online interactive software to help you fully understand all the electronic principles. For a complete product description and sample software, please visit our webstore.

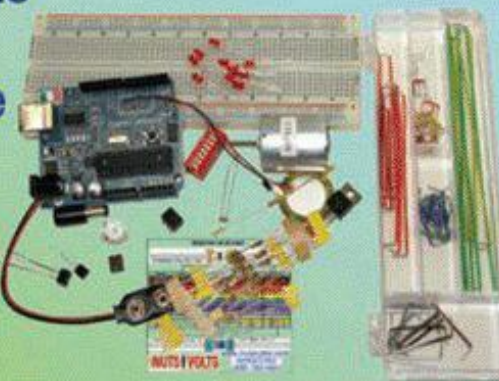
Regular Price \$79.95

Subscriber's Price \$75.95

SPECIAL OFFERS

An Arduino Workshop
Are you puzzled about the Arduino but finding it difficult to get all the pieces in one place?
Joe Pardue
SmileyMicros.com
Book \$44.95

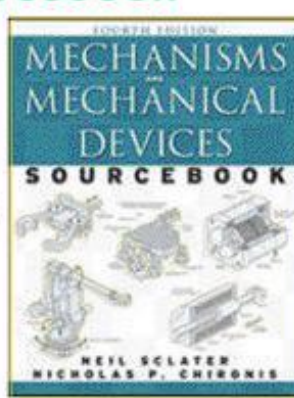
Puzzled by the Arduino?
Based on the *Nuts & Volts* Smileys Workshop, this set gives you all the pieces you need!
Book and Kit Combo \$124.95



Kit \$84.95

For more info on this and other great combos, please visit: <http://store.nutsvolts.com>

Mechanisms and Mechanical Devices Sourcebook
by Neil Sclater, Nicholas Chironis
Over 2,000 drawings make this sourcebook a gold mine of information for learning and innovating in mechanical design. Overviews of robotics, rapid prototyping, MEMS, and nanotechnology will get you up to speed on these cutting-edge technologies. Easy-to-read tutorial chapters on the basics of mechanisms and motion control will introduce those subjects to you. **Reg \$89.95 Sale Price \$69.95**



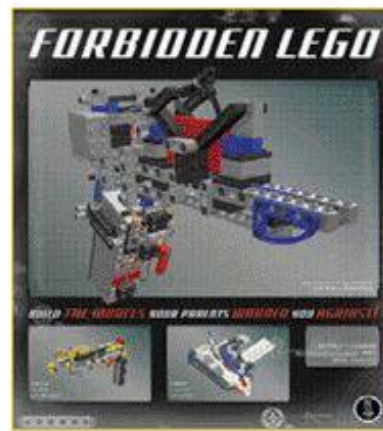
Enter the world of PICs & Programming with this great combo!



Combo Price \$175.95

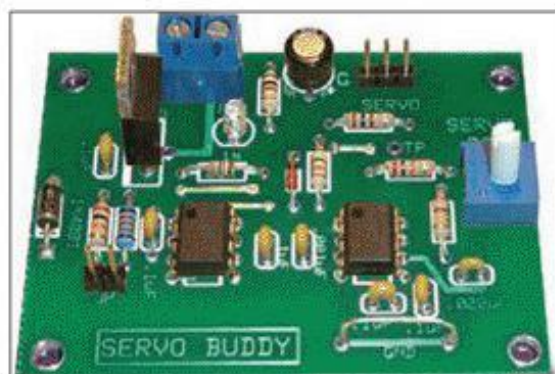
For complete details visit our webstore @ www.nutsvolts.com

Forbidden LEGO
by Ulrik Pilegaard / Mike Dooley
Forbidden LEGO introduces you to the type of free-style building that LEGO's master builders do for fun in the back room. Using LEGO bricks in combination with common household materials (from rubber bands and glue to plastic spoons and ping-pong balls) along with some very unorthodox building techniques, you'll learn to create working models that LEGO would never endorse. **Reg \$24.95 Sale Price \$19.95**



PROJECTS

The SERVO Buddy Kit



An inexpensive circuit you can build to control a servo without a microcontroller.



For more information, please check out the May 2008 issue or go to the SERVO webstore.

Includes an article reprint.
Subscriber's Price **\$39.55**
Non-Subscriber's Price **\$43.95**

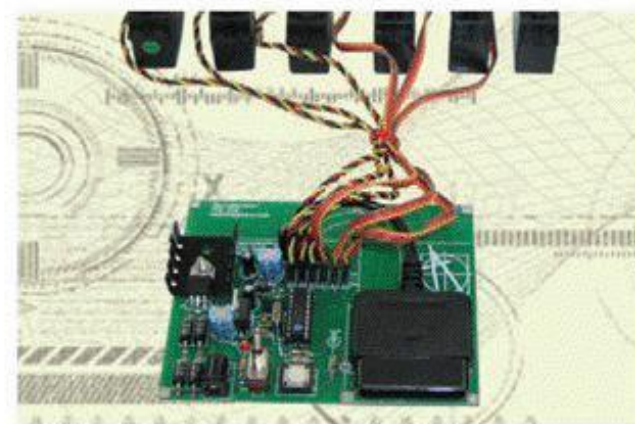
16-Bit Micro Experimenter Board



Ready to move on from eight-bit to 16-bit microcontrollers? Well, you're in luck! In the December 2009 *Nuts & Volts* issue, you're introduced to the 16-Bit Micro Experimenter. The kit comes with a CD-ROM that contains details on assembly, operation, as well as an assortment of ready-made applications. New applications will be added in upcoming months.

Subscriber's Price **\$55.95**
Non-Subscriber's Price **\$59.95**

PS2 Servomotor Controller Kit



This kit accompanied with your own PlayStation controller will allow you to control up to six servomotors. Includes all components and instruction manual.



For more information, please see the February 2011 edition of *SERVO Magazine*. Assembled units available!

Subscriber's Price **\$79.95**
Non-Subscriber's Price **\$84.95**

Twin Tweaks

BRYCE WOOLLEY & EVAN WOOLLEY



THIS MONTH:

Why Did The Robot Cross The Road?



THE CROSS-LINK CONTROL SYSTEM FROM CROSS THE ROAD ELECTRONICS.

This month, we have the pleasure of presenting the Cross-Link Control System from Cross The Road Electronics. Cross The Road — founded by FIRST Team #217 mentors Mike Copioli and Omar Zrien — aims to offer a control system ideal for the competitive robotics environment by providing an accessible and expandable system based on the Controller Area Network (CAN) protocol most often seen in automobiles. It just so happened that we had the perfect platform on which to test out the Cross-Link control system — Protobot, a project from our FIRST days of yore that has made the occasional appearance in our column (see the May '09 issue for an earlier brain transplant).

Cross The Road makes the bold claim that the Cross-Link system can be implemented by a user of any programming background, and that it can be used with any operating system and any programming language. Such broad applicability is meant to entice programming veterans, as well as novices. Finding the balance between accessibility and expandability is a difficult task, and one that is all the more critical in a competitive



THE 2CAN ETHERNET TO CAN GATEWAY.

robotics setting where team members may come and go, and roboticists of all skill levels need to be accommodated. And, just in case that balancing act isn't overtly exciting enough, the Cross The Road team also offers the uCANDrive app which allows users to configure and drive their Cross-Link robot over their Android smartphone.

Yes We CAN

The Cross-Link control system is comprised of three main components. The first of those is the 2CAN, an Ethernet to CAN gateway that allows your computer to communicate with the rest of the CAN modules in the system. The versatile Ethernet connection also allows users to implement any processor or tool suite with the Cross-Link system. The compact module includes two Ethernet connections, a CAN connection, and wires ready to be outfitted with the connector of your choice.

The other core member of the Cross-Link menagerie is the CANipede Robot Control Module (RCM). The CANipede has all of the inputs and outputs characteristic of a robot controller — including eight PWM channels, eight analog inputs, four digital I/Os, eight solenoid channels, four quadrature encoder channels, four relay outputs, and two CAN connectors. The CANipede includes two sockets for CAN connectors and a power cable with wires, once again waiting for the perfect connector for your unique project.

Even though a centipede doesn't necessarily conjure up imagery of hardiness, the CANipede has the robustness to match the CAN protocol. All of the inputs and outputs are short circuit protected. This will probably be a lifesaver for many teams, because as much as you try to avoid drilling

any new holes after the electronics have been placed, we know that the vagaries of fate may compel some last minute modifications. With the CANipede's short circuit protection, accidentally bridging some pins won't kill your robot, and the problem is even helpfully identified by the hardware LED status light.

The last major component of the Cross-Link control system is a portable wireless router from TRENDnet. The portable router operates on the 802.11n wireless standard, and it comes with an Ethernet cable, a USB power cable, and a wall power source.



THE TRENDNET TRAVEL ROUTER.



THE CANIPEDE ROBOT CONTROL MODULE.

A lot of folks have probably heard of CAN in the automotive context, where it has been the standard protocol since the 1980s. CAN is the standard in part because it is so effective in high noise environments. Cars put electronic equipment in close proximity to a noisy engine. Another realm that pits sensitive electronics close to noisy motors is competitive robotics which is one reason why the 2CAN and CANipede were developed. The Cross-Link system also aims to strike the balance between accessibility and expandability, and the first step in testing that balance would be to implement the system on an appropriate platform.

Our aforementioned Protobot was an ideal platform. Protobot was constructed in 2004 by Team #1079 from leftover parts from our FIRST kits from 2003 and 2004 (for more on Team #1079's 2004 season, you can dig deep into the *SERVO* archives for the September through December '04 issues). Using the basic frame pieces, two CIM motors, and a serious gear train, Protobot is a six-wheeled robot with two drive wheels. With a complete electrical skeleton including a power distribution block and a set of Victor 884s, Protobot would be a perfect way to test out the Cross-Link control system.

All of the software and documentation can be downloaded from the easily navigable Cross the Road website (www.crosstheroadelectronics.com). The documentation includes manuals for each individual component, schematics, and a manual for the implementation of the Cross-Link system as a whole. The only software required to implement the system is the CAN Firmware Utility, used to update the firmware of the 2CAN and CANipede, and the Robot Control Software (RCS) — a graphical interface that allows the user to control the robot over their computer.

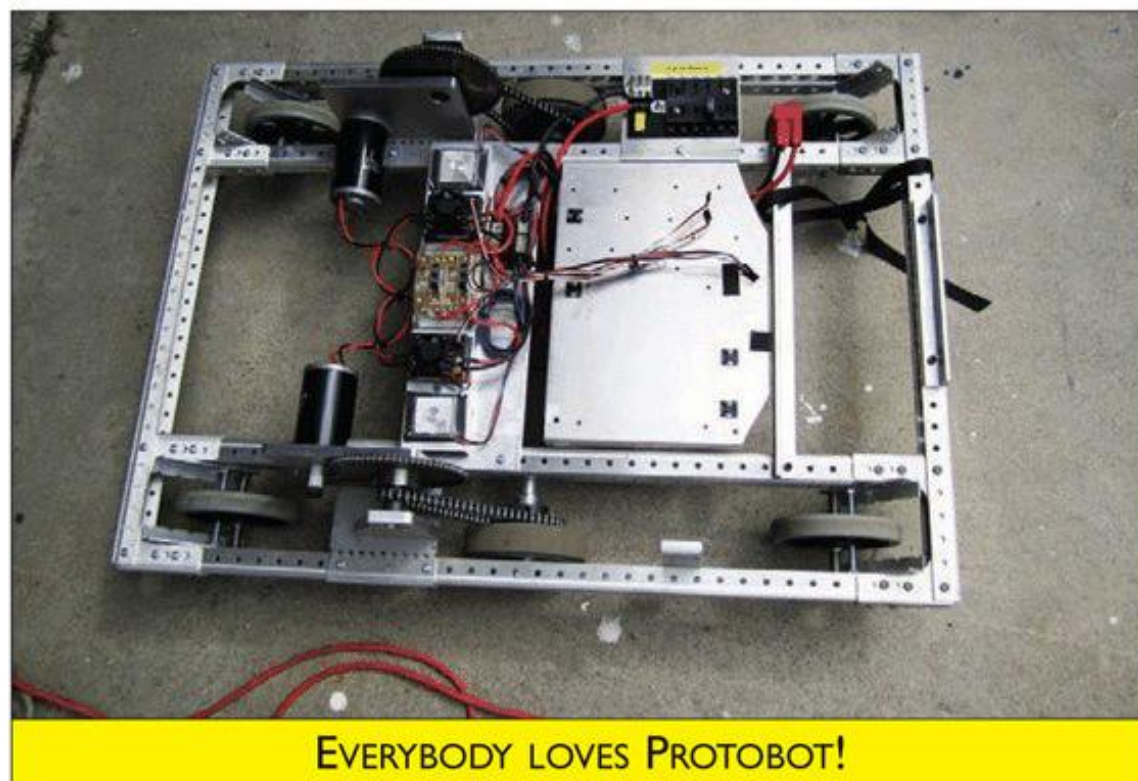
The Roaming Drone

The Cross-Link manual for the most part gives

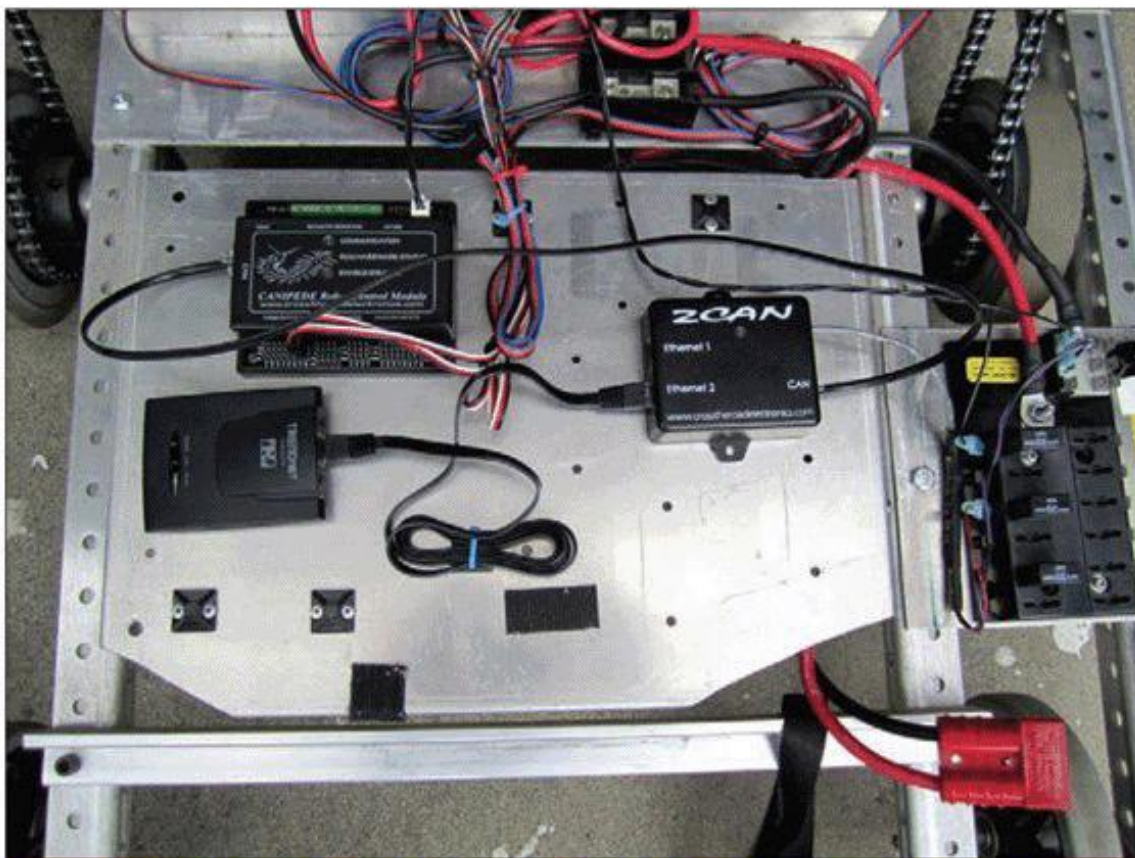
meticulous instructions on how to implement the control system — the first 10 pages are spent introducing the components of the system and walking you through the installation of the software. Such detailed instructions make the initial setup of the RCS fairly straightforward. One of the only brain ticklers that arose during the physical implementation of the system was sorting out the power requirements.

All components of the control system need independent power, because power is not transferred along the CAN or Ethernet cables. The 2CAN and CANipede modules are flexible, and will take power between 6.5 and 24V at 500 mA. The odd mod out is the TRENDnet router which demands nothing higher than 5.1V at 1A. For FIRST teams looking for a project in the off-season, meeting these power requirements is a simple affair. The power distribution board from distributor AndyMark will supply 12V and 5V (and 24V, for that matter) from a battery like the one supplied in the FIRST kit of parts.

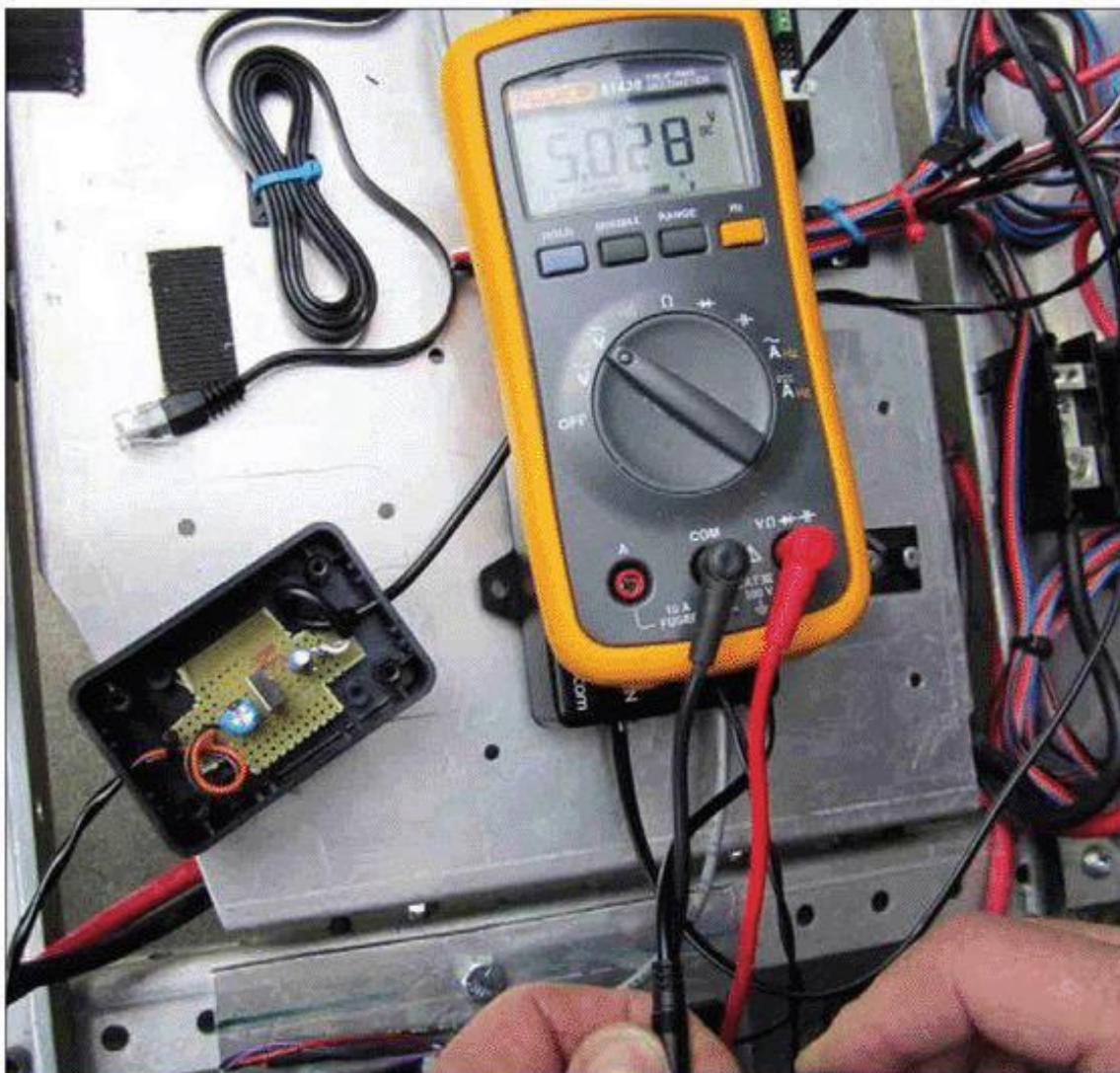
Protobot was made from FIRST parts from the 2004 season, and the power distribution block will simply source 12V with spots for breakers from our 12V battery. This met the requirements for the 2CAN and the CANipede, but we



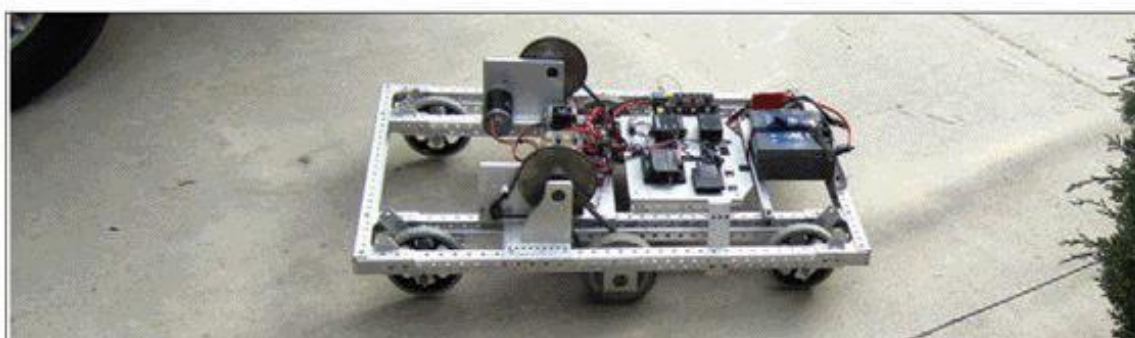
EVERYBODY LOVES PROTOBOT!



WIRING UP THE CONTROL SYSTEM FOR THE FIRST TIME.



CHECKING THE POWER FROM OUR REGULATOR.



DRIVING PROTOBOT WIRELESSLY.



had to get a little creative to power the TRENDnet router. The solution was a simple voltage regulator circuit using a 7805 IC and a couple of capacitors. Because of the strict warnings that came with the router that any voltage over 5.1V could cause irreparable damage, we thought it best to check our circuit with the infinitely useful multimeter before hooking everything together. Sure enough, the regulated voltage was in the Goldilocks zone with an output of 5.028V. We couldn't have gotten a better deal had the Priceline negotiator advised us that we could name our own voltage!

Physically wiring up the components was straightforward. The 2CAN and CANipede come with free wires that were easily crimped into the sockets necessary to connect to the power distribution block. The TRENDnet router comes with a USB power cable that allows the unit to be powered from the USB port on a computer. We cut the end of the cable that attached to the router and extended the wires on the other side, so they could comfortably accommodate the required sockets. The compact components were easily mounted on an electronics shelf on Protobot, and a few strategically placed strips of Velcro were able to keep everything sitting tight.

The manual sagely suggests a bench test to make sure everything is communicating properly. The first step is to make sure that the 2CAN and the CANipede have updated firmware which can be done with the 2CAN Firmware Utility. The Firmware Utility has a fairly self-explanatory user interface that gives helpful feedback as you go through the steps of establishing communication with the 2CAN. The only part of this process that was a hassle was synching the IP address of the computer with the 2CAN. The Firmware Utility ostensibly gives you a way to change the IP address of your computer through the utility itself, but the changes never seemed to take to our computer. Thankfully, the manual also walks users through how to change the IP address of their computer manually, which worked like a charm.

After updating the 2CAN, you can open the Web Dashboard to update the CANipede. The Web Dashboard can be used to get helpful feedback from the entire control system, with the CANipede being just one possible node. Other nodes can be easily wired in parallel, but initial implementation of the system does not require anything so involved. Our only caveat about loading the firmware is to be cognizant of the jumper on the CANipede. The jumper can set the CANipede to termination resistor or bootloader mode. For the purpose of updating the firmware, make sure the CANipede is in bootloader mode.

After updating the firmware, the robot is ready for the first bench test. The bench test uses the RCS to activate a solenoid switch. No actual solenoids were harmed (or used) in the making of this bench test, because the CANipede has LEDs that light up when each solenoid switch is activated. Using a USB game controller like the dual joystick model from Logitech is recommended. We didn't have one on hand, but they can be picked up at your local electronics store

for only about \$15.

Mapping the controls to the Logitech controller is very easy, because all you have to do is navigate through a few drop-down menus. The RCS is even accommodating to those without fancy USB controllers, because you can set the inputs to be buttons and sliders that will appear onscreen.

We were very happy with Protobot's new control system. The controls were a breeze to map to the game controller, and we were soon driving Protobot around our driveway just like in the glory days of yore. The handheld game controller is a lot more manageable than the controller boards we built for our FIRST robots, and the RCS was constantly giving helpful feedback like the robot voltage and a graphical meter that tracked our input via the joystick. The various buttons and triggers on the game controller also provided ample opportunities to manipulate any mechanisms that could possibly go on a FIRST robot.

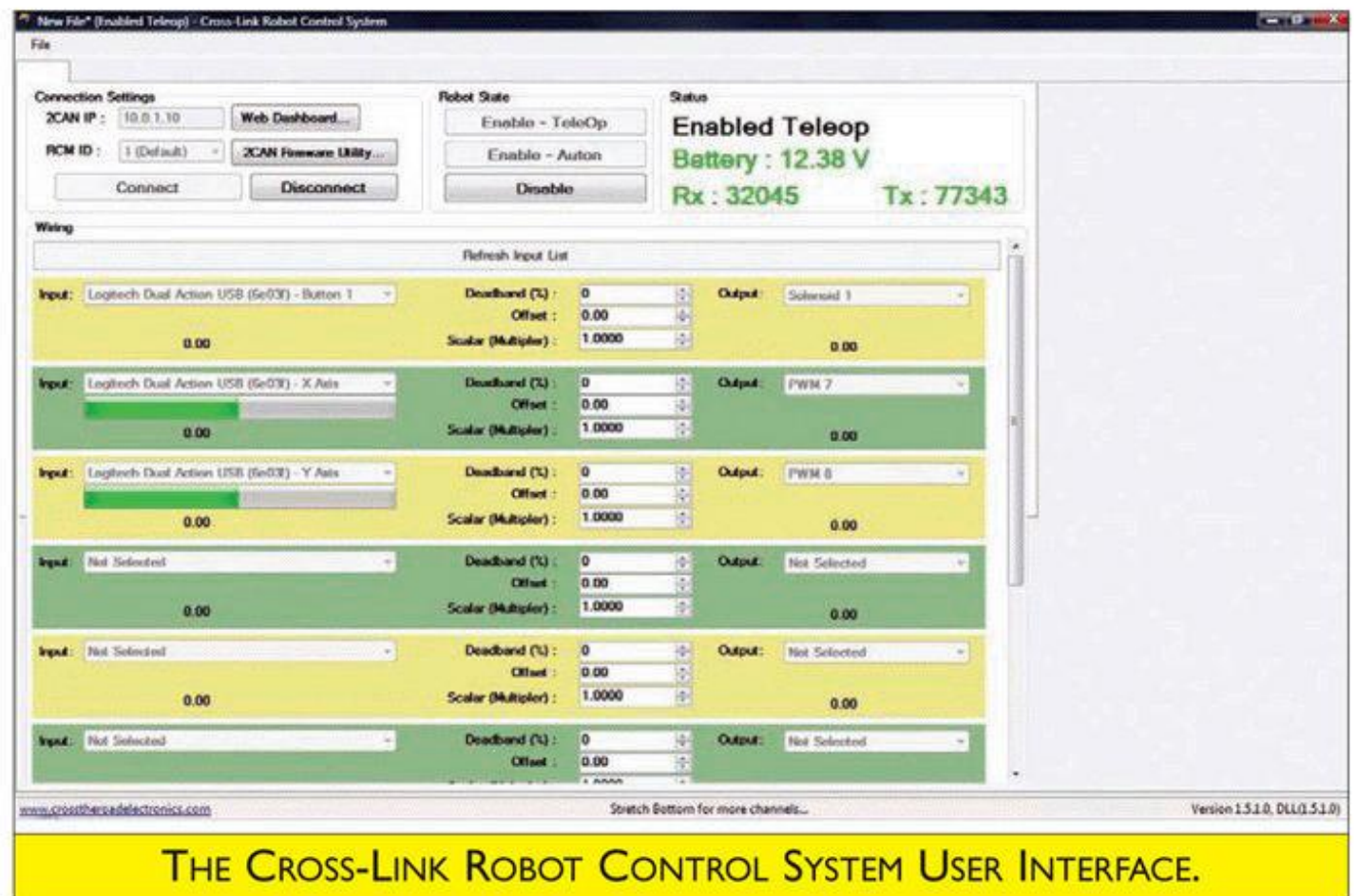
To The Victor, Go The Jaguars

The system worked well and we were happy with our sleek game controller, but one area for improvement jumped out at us. One of the cool aspects of the Cross-Link control system was the advantages associated with the CAN protocol. Those advantages of noise immunity were not exploited to their full potential, because the entire system was not on the CAN network. The biggest exclusions were our Victor speed controllers which used noise susceptible PWM signals. This was quite the disappointment, because fine motor control could be potentially disrupted by a noisy robot.

The disappointment, however, was easily remedied. Even though the scrappy Victor 884 will always have a place in our hearts (and our combat robots), a newcomer has pounced onto the scene that promises to take full advantage of the CAN protocol offered by the Cross-Link control system. The feisty newcomer is the MDL-BDC24 Black Jaguar from Texas Instruments.

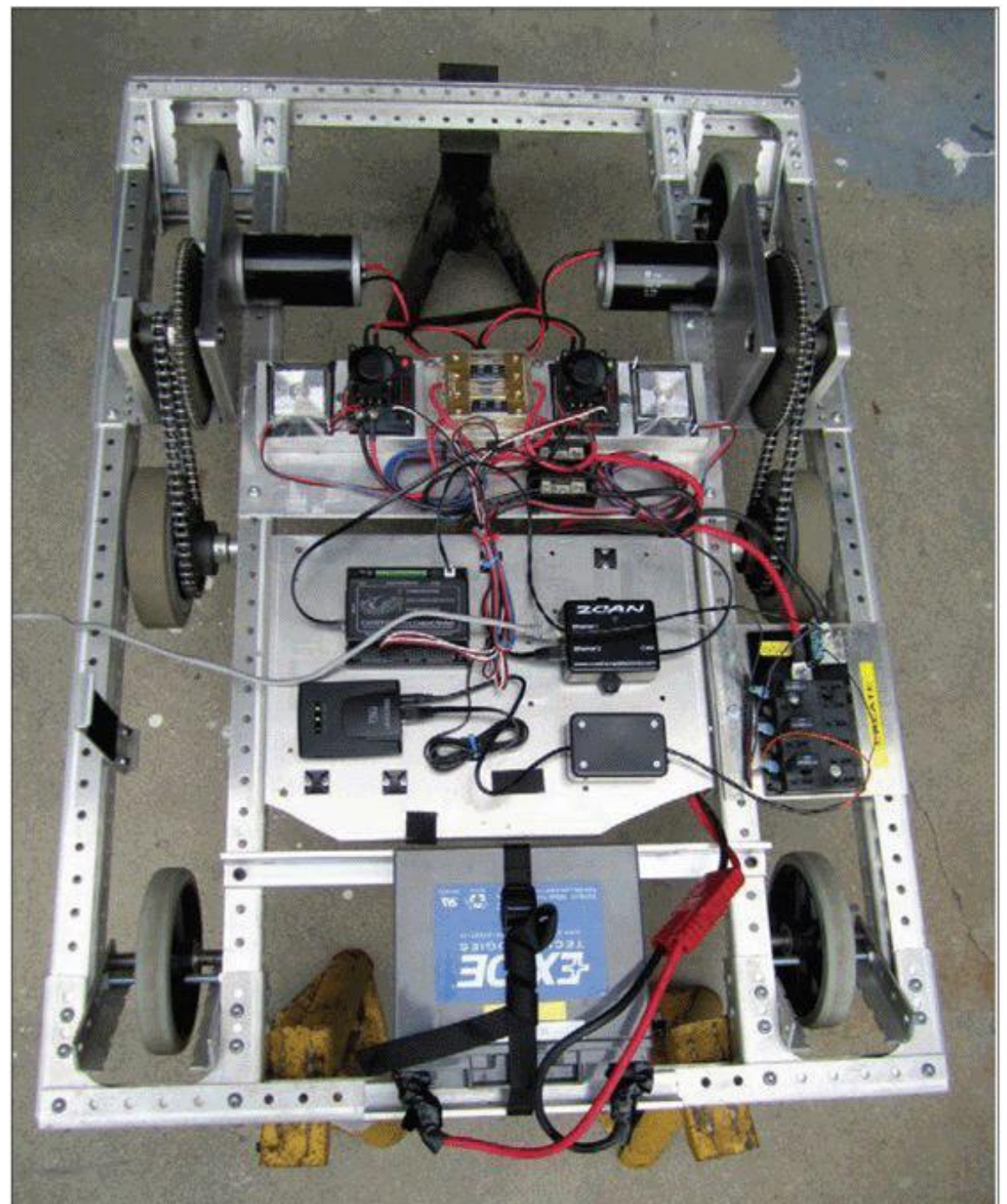
The Jaguar is a motor controller designed specifically for the FIRST competition – something for use in harsh, high noise environments while still being accessible to newcomers. The Jaguar was a part of the 2011 FIRST kit of parts, so the competitive environment and accessibility of the unit already seemed like a perfect fit for the Cross-Link control system.

Even more perfect is the fact that the Jaguar includes a CAN interface. This means it can be incorporated seamlessly into the Cross-Link system while maintaining the noise immunity of a CAN network. The only caveat we have about the impressive Jaguars is that they are quite a bit bigger than Victor 884s, but they still fit perfectly onto Protobot's accommodating frame.



Cool Factor Engineering

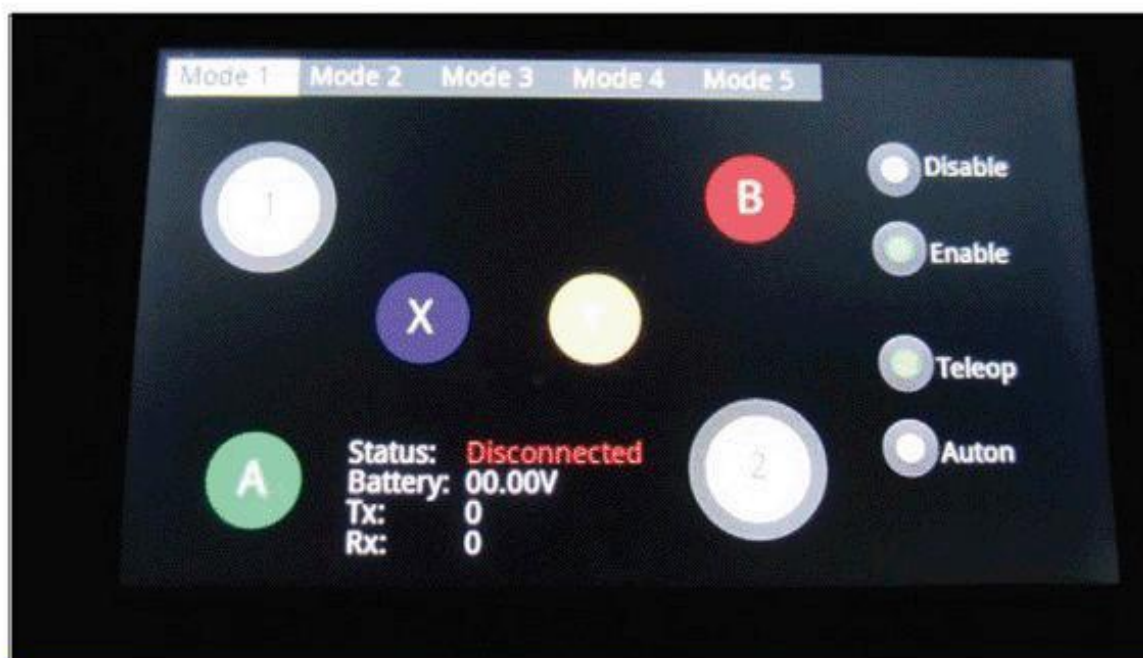
We have always been big fans of the cool factor. The cool factor can take a variety of forms. It may be a completely nonfunctional embellishment, like whimsical decals or a paint job. It may be functional, but perhaps



Twin Tweaks ...



JAGUARS TO HELP PROBOT REALLY PURR
(NOT THAT JAGUAR!).



THE UCANDRIVE INTERFACE.



ANDROID + PROBOT = AWESOME!

overdone to the point of hyperbole (like putting a large 3/4" thick 7075 aluminum plate on the front of your FIRST robot). The cool factor is anything that you can add to your project that will make people look at it and spontaneously utter "that's cool." Whatever form it takes, we think the cool factor is, in fact, an important part of any robotics endeavor. The cool factor helps people take ownership of the project. It gives them something to be proud of. Of course, any functional robot truly is something to be proud of, but a functional robot emblazoned with your own personality is something that cultivates a real sense of

personal accomplishment.

The folks at Cross The Road Electronics apparently feel the same way about the cool factor as we do, because in addition to the workhorse-like user interface of the RCS, users can drive their robot through the uCANDrive app for the Android smartphone. The uCANDrive app — which can be downloaded for free from the Cross The Road website or the Android app market — comes with five modes that allow for a broad range of control using joysticks, buttons, sliders, and even control for an onboard accelerometer. Setting up the uCANDrive is very similar to the setup with the RCS, and the wide array of possible buttons and sliders would be more than enough to control even the most intricate FIRST robot.

Using the Android touch screen to control the robot makes you feel like some sort of mechanical maestro manipulating the spatial operating interface from *Minority Report*. And much like a skilled maestro, users do maintain a degree of control over the sleek interface with the ability to map outputs and modify scaling and dead bands.

Controlling the robot with a flick of a finger on the Android touch screen was undeniably fun. The controls were responsive, and the sleek smartphone was even more compact than the USB game controller. The Android also had the advantage of being the wireless access point itself — the USB controller still needed to be hooked into a computer, but the driver could follow their robot around with the Android all day.

Far from being a gimmick, we think the uCANDrive app is a great way to demonstrate to students, in particular, the power of programming, the excitement of integrating technologies, and that their Android can, in fact, be used for something more important than Angry Birds.

When Control Systems Compete, You Win!

Reviving Probot gave us an idea for the perfect application for a control system like Cross-Link. We built Probot in 2004, after the 2004 FIRST season and in preparation for the 2005 season. Our FIRST team was at a crossroads. After two solid seasons, the original team members had only one more year before graduation. New members were coming in, but we needed a way to impart the enthusiasm that we had built up through two years of competition. That enthusiasm was at a critical mass after the 2004 competition, with the team brimming with ideas for next year and eager for a system on which to test them out. We built Probot as a way to test new ideas and demystify the robot building process for new members.

This sort of project would be a perfect opportunity to implement something like the Cross-Link control system. We think it's a great idea to have more choices when it comes to control systems for a robot, particularly in an environment like FIRST (or any competitive robotics, for that matter). Choosing between different control systems demands that teams understand what exactly makes those

systems different. Since a major difference here is the protocol, the simple availability of the CANipede RCM is a fantastic teaching moment on that core concept. The simplicity of implementing the Cross-Link control system is also a great way to achieve a major goal of a robotics team in transition: instilling confidence in the new team members.

One of our biggest challenges when recruiting new team members was to convince those with no experience that they too could build a robot. Getting acquainted with the components of the Cross-Link system and removing the additional intimidator of programming is a great way to demonstrate that robots are not an inscrutable collection of wires and circuit boards, but rather a logical structure of electronics. The depth of the system also allows veterans to sharpen their programming skills or add a variety of mechanisms with the assurance that they'll be able to control them with the tap of a button or manipulation of a slider.

Intrepid programmers have a lot to look forward to with the Cross Link RCS, because the entire project is open source. The code is all available online, and can easily be modified with an IDE like MPLAB. The inputs and outputs of the CANipede can be retooled, and the RCS interface can be modified. The 2CAN is the only closed source component of the system, but that shouldn't be too discouraging.

Future versions of the 2CAN firmware will even allow users to add their own custom web pages to supplement or take the place of the Web Dashboard. The entire Cross-Link system — which can be obtained through trusted FIRST distributor AndyMark — costs \$399.99, which should be well within reach even for smaller FIRST teams and other competitive roboticists.

The folks at Cross the Road Electronics envision a

```

namespace CrossLinkControlSystem
{
    class RobotController
    {
        UInt32 _handle = 0;
        int _rcmNodeId = 1;

        DLL_ctr_error_t _lastError = DLL_ctr_error_s_ctr_ok;
        bool _isConnected;
        DLL_CompState _comState;

        public RobotController()
        {
            _isConnected = false;
            _comState = new DLL_CompState();
        }

        public void SetRcmNodeId(int nodeId)
        {
            if (nodeId > 0) // must be nonzero
                _rcmNodeId = nodeId;
        }

        public int GetRcmNodeId()
        {
            return _rcmNodeId;
        }

        DLL_ctr_error_t HandleReturn(DLL_ctr_error_t ex)
        {
            _lastError = ex;
            if (_lastError != DLL_ctr_error_s_ctr_ok)
            {
                //TODO present error codes in status bar
            }
            return ex;
        }

        public bool Connect(string sIp)
        {
            if (_isConnected)
                Disconnect();

            if (HandleReturn(DLL_CTR_Init(sIp, ref _handle)) == DLL_ctr_error_s_ctr_ok)
            {
                _isConnected = true;
            }
        }
    }
}
    
```

TINKERING WITH THE CODE.

system that welcomes new family members every year. With an inner circle that already includes the 2CAN, CANipede, and friends like the Jaguar, we can only imagine how new additions would exponentially increase the possibilities of the Cross-Link control system. **SV**

Build Your Robot The Way YOU Want!

With the Cross-link Robot Control System!

With this system, there's no need to write software! Plus, you get portability, connectivity, and reliability for one low cost! You can use ANY processor, ANY platform with ANY toolsuite, or run it directly from a PC or laptop!



The kit includes:

- 2CAN Ethernet Gateway
- CANipede Robot Control Module (RCM)
- TRENDnet N Pocket Router
- Cross-link/CAN Cables

RECOMMENDED WEBSITES

<http://crosstheroadelectronics.com>
www.andymark.com
<http://code.google.com/p/crosslink-rcs>

SPECIAL THANKS TO

Mike Copioli from Cross the Road Electronics.

Andy Baker from AndyMark for the purr-fect Jaguars.

Paul Copioli from VEX for the infinitely useful Victors.



Cross the Road Electronics, LLC

www.crosstheroadelectronics.com



Then and NOW

NEW APPROACHES TO ROBOTICS EDUCATION

by Tom Carroll

At the end of March, I decided to go to the Autodesk Oregon Regional FIRST Robotics Competition in Portland to observe, photograph, and help out in any way I could. I've always enjoyed robotics competitions over the past few decades, most starting out with just a few competitors and some basic rules. FIRST — For Inspiration and Recognition of Science and Technology — started out the same way in a high school gym in Manchester, NH in 1991 and was (and still is) the brainchild of innovator Dean Kamen and MIT professor, Woodie Flowers. Before delving into the modern FIRST competitions, I'd like to go back a few decades to those classes at MIT, Professor Flowers, and the innovative student rivalry that started it all.

MIT Found a Better Way

Most of us have had a college professor who just seemed to use a better method to teach. Instead of a boring spiel that could have been presented by a tape recorder, he or she had that special talent of being able to tune into what the students were thinking and was able to present course material in a most unique way. I had such a professor who taught anatomy and physiology — the absolute hardest class that I'd ever taken, but the most interesting, by far.

I remember to this day, many of the most obscure details of the body, organ function and structure, and numerous Latin names of all the bones in our bodies. I think that's pretty amazing for a person who ended up in aerospace, robotics, and RF systems, and had nothing to do with medicine or human physiology. This learning/retention was not because I was an amazing student. I give credit to this one professor because he taught in a way that made me and my classmates want to learn.

Putting Mettle to the Metal

MIT Professor Woodie Flowers (**Figure 1**) is one of those amazing teachers. He taught the Mechanical

Engineering course, "Introduction to Design" back in the 1970's; it was recently renamed to "Introduction to Design and Manufacturing." This was a hands-on class; not one with a professor who droned on as the students nodded off. It was an amazing success and was the seed for contests to come later.

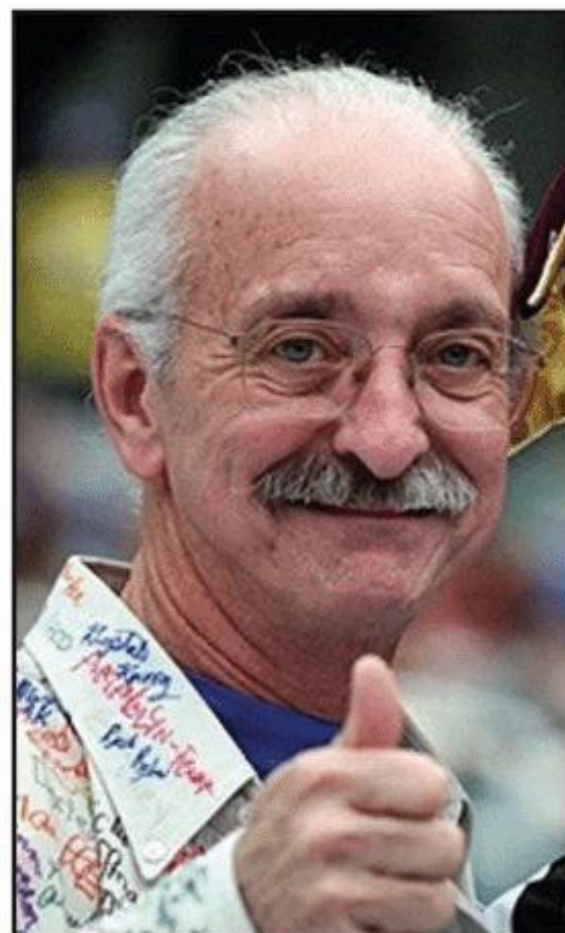


FIGURE 1. Woodie Flowers

The original syllabus for Flowers' course specified as a final assignment, "Design and build a robotic system for putting a round peg in a square hole, while a competitor tries to put another peg into the same hole." Not only did the student have to design a pretty robust robotic system, the student was in a competition with other students — always an exciting plus. Students received a box of materials that included two motors, various sprockets, cords, cardboard tubes, cardboard, and even some rubber bands. Unlike a simple term paper that some educators might request, Woodie had his students put 'mettle to the metal' and produce a working robot. He then had them compete with fellow student's machines. Like the anatomy and physiology course that I took years back, this course was not an easy A, but the

students loved it.

After the students had formed teams and completed their individual robots, they took their creations to the on-campus Johnson Athletic Center. One would think it was some sort of rock concert turned athletic event inside as

hundreds of students packed the center, with foot stomping, clapping, and cheering for various entrants as they clawed their way to various goals with balls or blocks in their maw.

At the final competition, cheers rang out for the most outrageous robot or even the most spectacular failure. The major difference in this competition was that the entrants were not the students, but their robotic creations. These events were, however, much more noteworthy iconic classics that typified what has made MIT a world-renown institution of basic and advanced science and engineering. A scene from the 2008 competition is shown in **Figure 2**.

The rules and goals changed each year, as well as the type of robot that each team constructed. These competitions became so noteworthy that they were frequently televised over network and public television stations. Other schools and universities soon learned that using robotic devices not only taught the many facets of engineering in a hands-on manner, but the competitive aspect honed the individual student's ability to work within a team to achieve the goal of the particular contest.

Since 1970, Flowers' hands-on course has taught gracious professionalism and has inspired many students to go into the engineering field as first-rate engineers in companies across the nation. No field of science course would be complete without the accompanying lab sessions. However, labs in the past followed a rigid outline geared to the textbook. This innovative approach has now been adopted by many universities with a similar strategy in teaching technical subjects.

Worcester Polytechnic Institute's Robotics Curriculum

Earlier this year, I participated in an online NextGen Education and Research Robotics Virtual Summit presented by Robotics Trends' Virtual Conference Series. Among the many presenters who gave some very interesting talks and the virtual booths who shared some interesting approaches to education, there was a talk and booth by Worcester Polytechnic Institute (WPI) in Massachusetts. Technical and community colleges have offered courses in factory robot implementation, maintenance, and operation for a number of years. MIT, Stanford, and Carnegie Mellon, as well as many other universities have long had courses leading to advanced degrees in mechanical, computer, and electrical engineering with strong emphasis upon robotics. However, there are few actual degrees in robotics or robotics engineering.

WPI feels that their series of Robotics Engineering courses sets the way for those who desire to design and manufacture robots for the many fields of this relatively new science. As they state, "The robotics revolution is underway, and a new breed of engineers is needed to face the challenges that this exciting field represents. WPI, the leader in project based education, continues its pioneering tradition by developing the nation's first Bachelor's degree program in Robotics Engineering. In addition, WPI also offers a

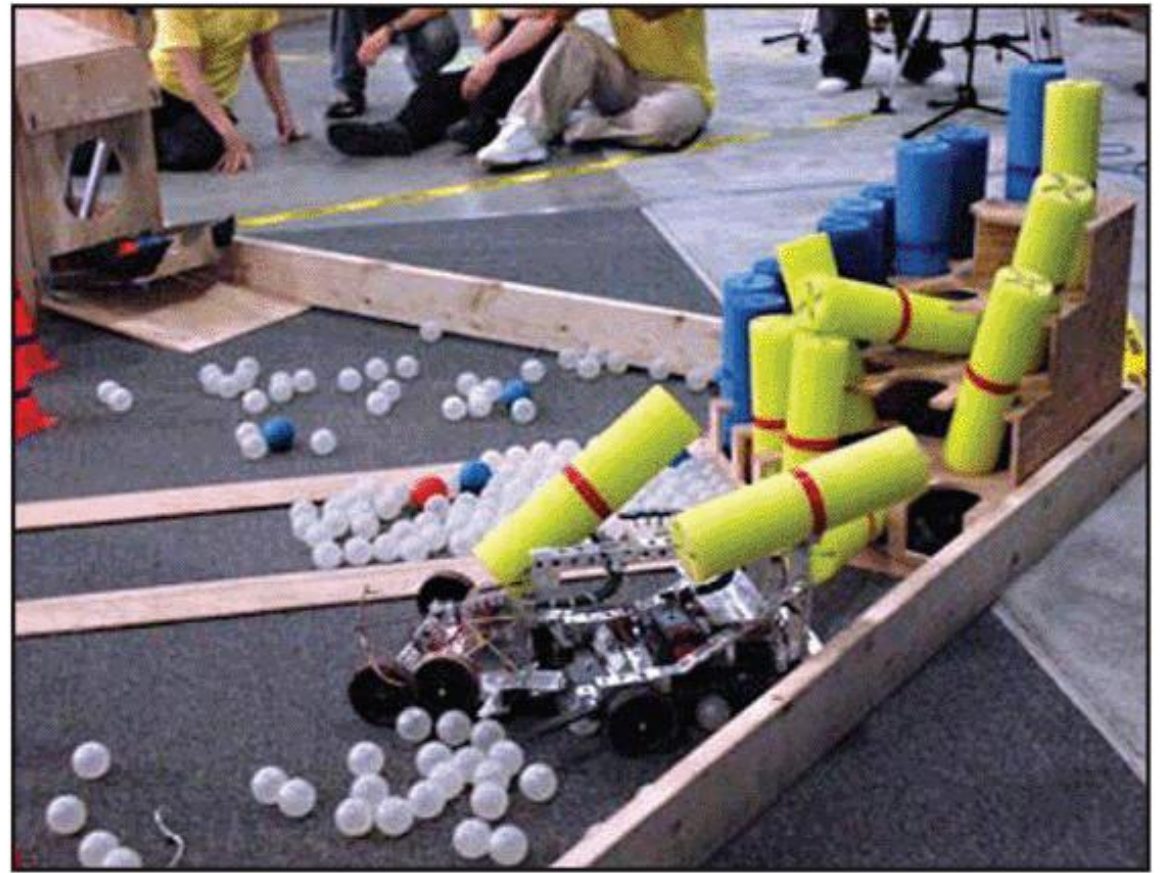


FIGURE 2. The 2008 MIT ME 2.007 competition.

Master's and Ph.D. in Robotics Engineering, with research in sensing, control, manipulation, learning, interaction, and medicine."

WPI Robotics Team Took First Place in 2009 NASA Competition

In the fall of 2009, the WPI sponsored robotics team took home first place in NASA's 2009 Regolith Excavation Challenge at the NASA Ames Research Center in Mountain View, CA. The challenge was to develop and demonstrate a robotic device that could dig up simulated moon dust/soil and deposit at least 150 kg of the material in a bin in 30 minutes or less. There were three prizes: \$500,000 for first (WPI); \$150,000 for second; and \$100,000 for third. The WPI team, headed by Paul Ventimiglia — a WPI robotics engineering major, beat 22 other teams by collecting and depositing 439 kg of regolith in the collection bin.

Moonraker 2.0 shown in **Figure 3** was built by Ventimiglia's team — Paul's Robotics — that also consisted of

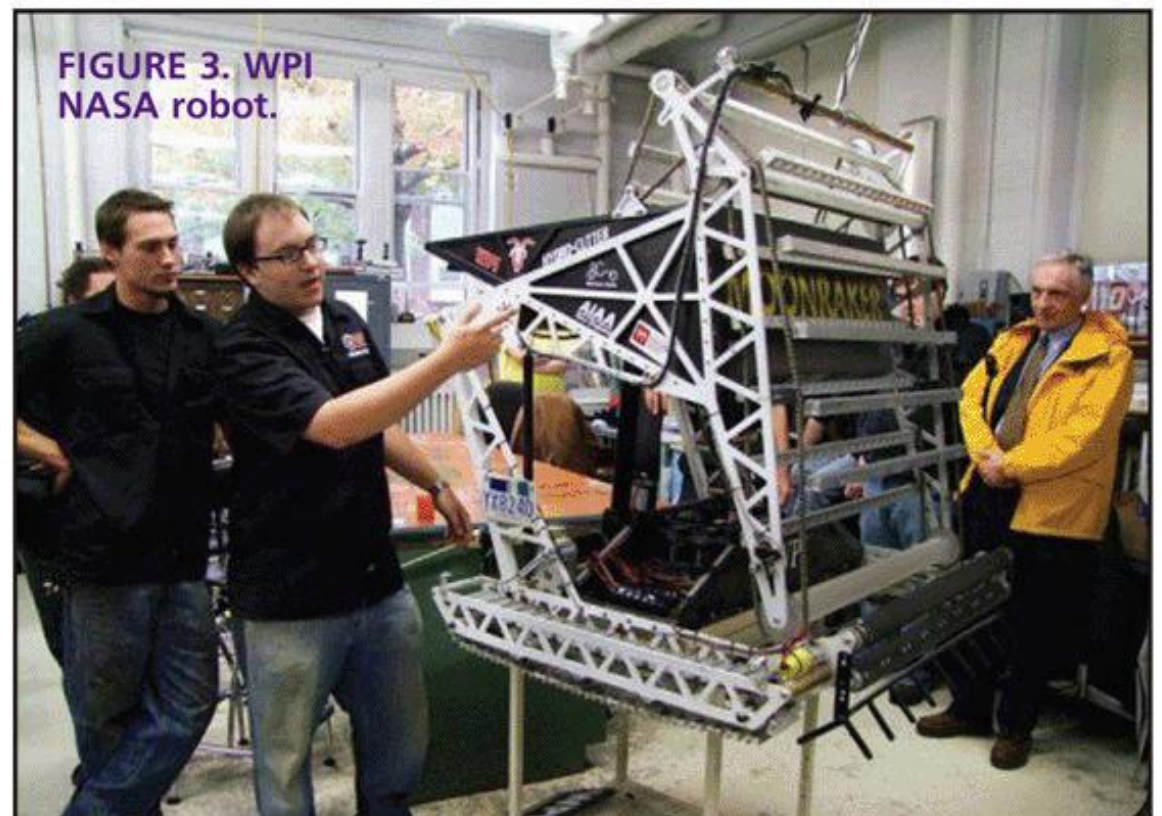


FIGURE 3. WPI NASA robot.

FIGURE 4. Hero 2000.

WPI faculty, staff, students, and alumni, and was part of NASA's Centennial Challenges Program that was formed to help inspire innovative solutions to technical challenges in the aerospace industry.

Moonraker features a series of scoops that continually rotate to collect the lunar soil. Once the robot's bin is full, the team remotely navigates it to the collection bin and deposits the regolith by raising the collector arm. NASA specifies that the competing robots must be battery operated, weigh less than 80 kg, and fit within a 1.3 meter cylinder. NASA further specifies that the robots must also employ only technology that could be used on the moon. Considering that the lunar soil is really the only natural resource available to future lunar pioneers, the use of this material is necessary to produce building materials, air, and water.

Teaching Materials for Robotics Classes

About four years ago, I discussed the Heath Hero 1 and Hero 2000 robots shown in **Figure 4** and associated courses. In the '80s, courses such as these were about the only way a person could gain

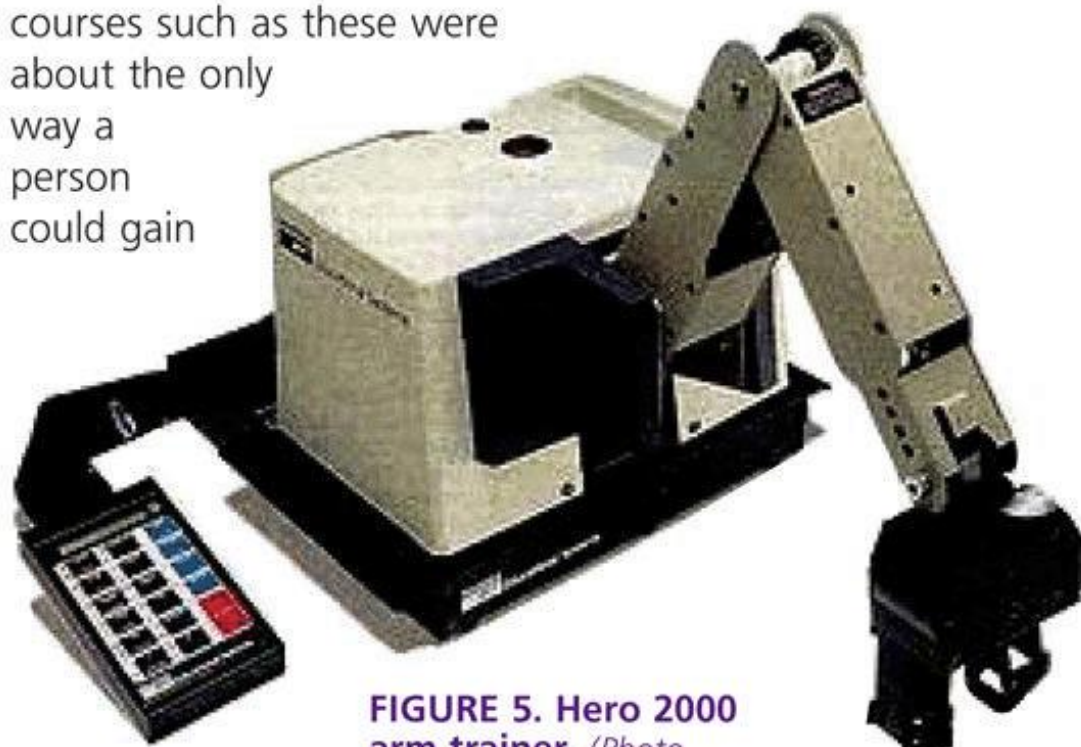


FIGURE 5. Hero 2000 arm trainer. (Photo courtesy of theoldrobots.com.)

knowledge about this brand new field as there were virtually no robotics courses in higher level institutions. Many community colleges eagerly used these robots as teaching aids as the core of electronics classes. The Heathkit Robotics and Industrial Electronics course was pretty thorough. In fact, I still have the manuals. Even though most preferred the arm on the mobile base, Heath sold an arm trainer (shown in **Figure 5**) that advanced classes could use for teaching basic industrial arm kinematics. There were a few other educational robots available in those days, such as the Microbot Teachmover shown in **Figure 6**.

Robotics Education for Home and Classroom Study

Compared with the '80s, today's assortment of classroom materials for teachers of robotics at all levels is almost unlimited. Most of the educational materials currently available consist of complete mobile robots or mobile bases that students can adapt with sensor arrays, arms, or other devices. Costs for instructional materials can be less than \$10 to hundreds of dollars per student. Several

years ago when I lived on Orcas Island off the coast of Washington State, I saw some robot kits made by McGraw Hill at a discount bookseller for less than \$10 that were originally \$59.95 each. The robots were based on the Microchip

PIC16C505

microcontroller (**Figure 7**). I

ordered a couple dozen of these

"Build Your Own Robot Kits" to teach a robotics class. It really doesn't take much to help a student realize that engineering, science, and robotics might lead to a rewarding career.

There are dozens of quality robotics courses that are applicable to both home study and the classroom. Some use a mobile base as I mentioned earlier, or have a student build a series of different robots ranging from cute dogs to multi-legged walkers to humanoids. Multi-legged or humanoid robots are generally expensive due to the number of servos required to drive each degree of freedom or axis, and can cost well over \$1,000. One of the most available robotics study series is the Parallax BoeBot kits (I covered these last month). The BoeBot shown in **Figure 8** is a mobile robot system starting at about \$150 and is based on the popular BASIC Stamp series of microcontrollers and the Board of

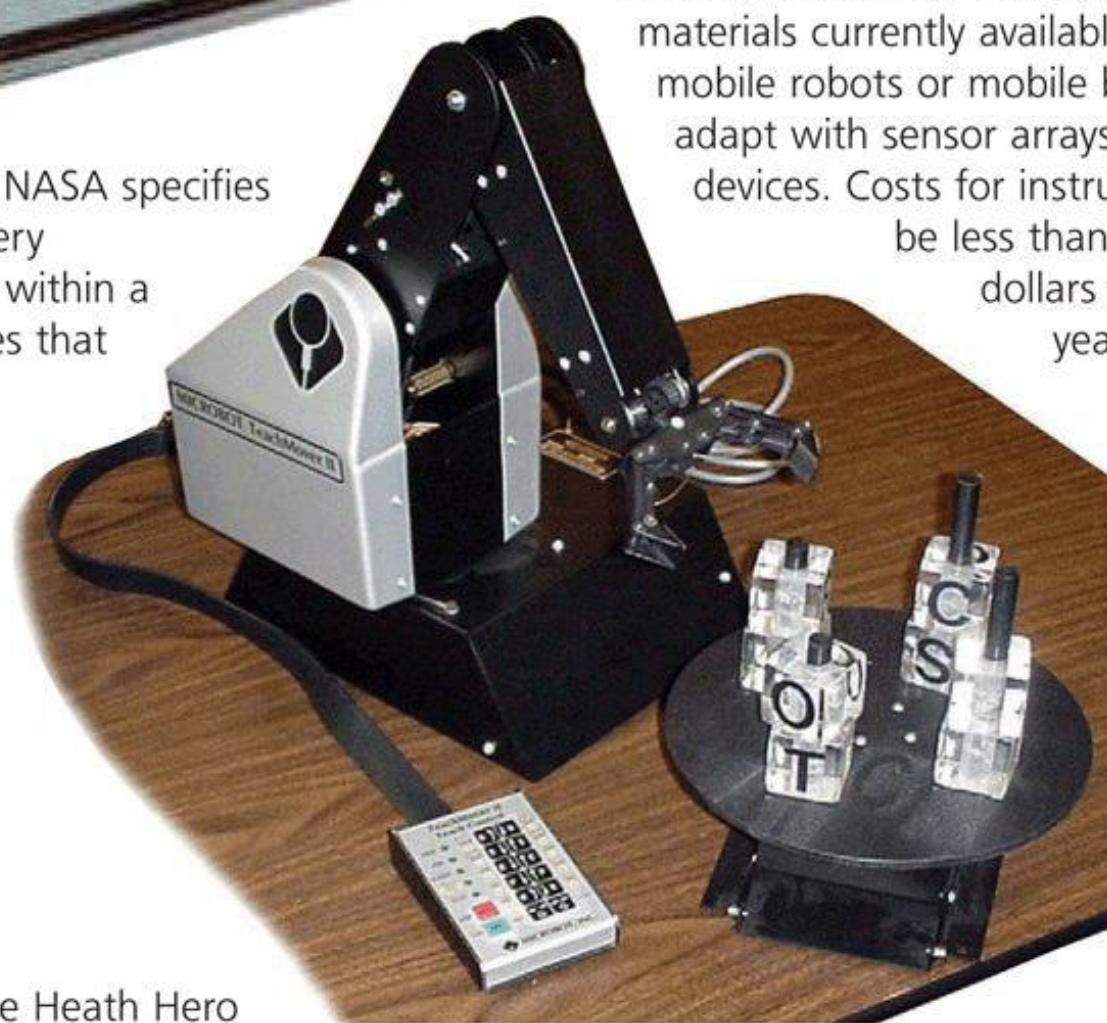


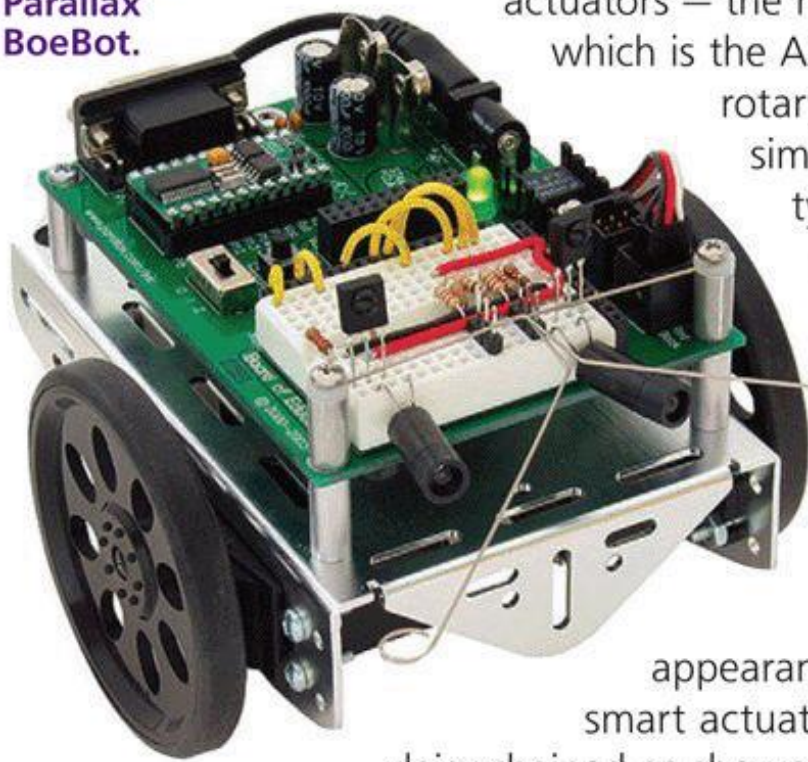
FIGURE 6. Microbot Teachmover.

Education experimenter's board. These kits have found their way into homes, as well as classrooms around the world. Parallax has many kits and variations available for educators.

The other tour de force in robotics education (and one of the first educational robot kit manufacturers) is the series of LEGO kits. The LEGO Mindstorms NXT 2.0 is the latest kit (shown in **Figure 9**) that features the powerful NXT microprocessor brick used with all their robots. This brick features a 32-bit processor with a large matrix display and four input and three output ports with Bluetooth and USB communications. The software has been greatly improved and the three servos, and two touch, ultrasonic distancing, and a color sensor complement the instruction manual to allow for classroom instruction, as well as a student learning on their own.

The Korean company, Robotis, is also a player in the US educational field with a series of robotic kits based on their Dynamixel

FIGURE 8.
Parallax
BoeBot.



actuators — the most popular of which is the AX-12. These rotary actuators are similar to the many types of model aircraft servos that have

been used for years, but only in

appearance. These smart actuators can be

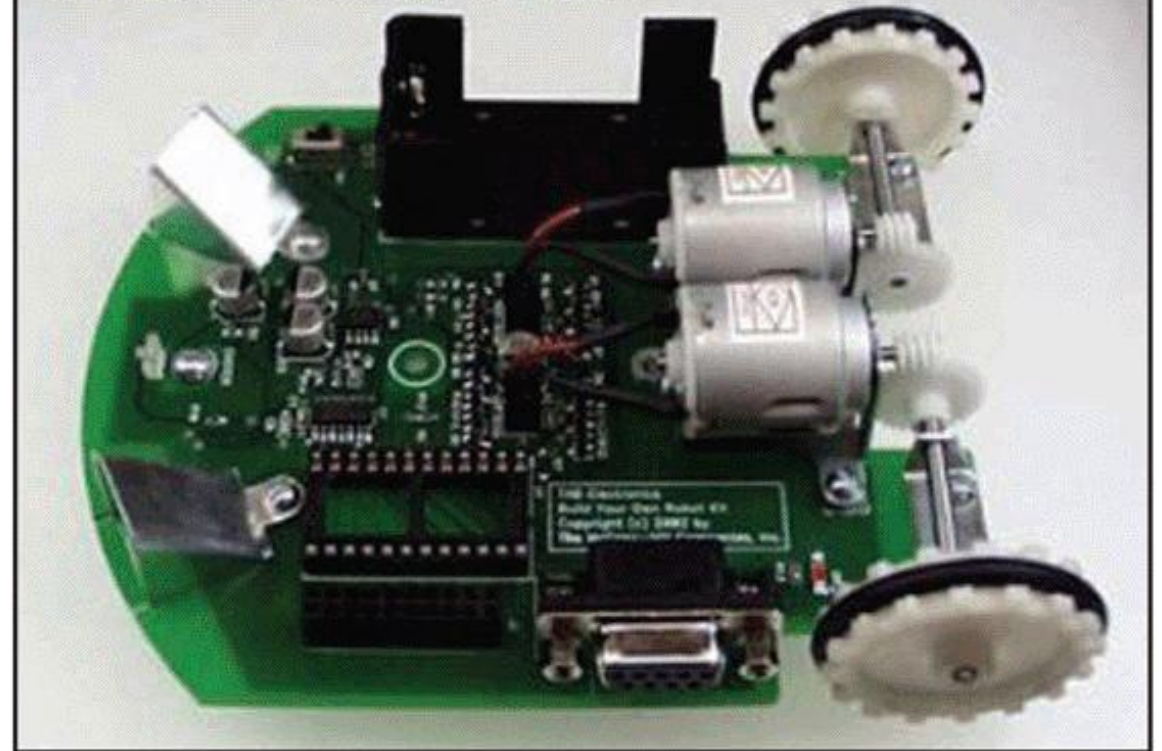
daisy-chained as shown in

Figure 10, and rely only on a single

serial address bus, power, and ground for any number of servos. The uniqueness of these actuators allows educators and students to use them in many configurations. Robotis' educational line begins with the OLLO Explorer kits for younger children then moves up to the many variations of the Bioloid kits and the more advanced Darwin-OP Open Platform Humanoid Project.

Another kit is the Robotech Robodesigner Educational Robot Kit out of Japan. The basic two-level robot (shown in **Figure 11**) is available for \$189.95, and offers endless robot configurations for classroom and individual study. This programmable robot kit is ideally suited for educational instruction for middle schools through university level classes. All basic parts are included in the kit. As Robotech states, "The RDS-X01 comes highly recommended by the Robot P.E.T.S. team due to its endless configuration options, sensors with line tracking capabilities, user friendly drag and drop programming interface, and optional

FIGURE 7. Tab-McGraw Hill robot.



teacher's manual."

STEM Education Coalition

STEM is the latest acronym in today's educational sector. Standing for Science, Technology, Engineering, and Mathematics, the STEM Education Coalition works to support STEM programs for teachers and students at the US Department of Education, the National Science Foundation, and other agencies that offer STEM related programs. The US has managed to slip behind many other countries in the world with the number of students graduating in the fields of science and math.

FIRST — For Inspiration and Recognition of Science and Technology

My visit to the Autodesk Oregon Regional FIRST Robotics Competition in Portland was everything that I expected and more. Just as in Woodie

Flowers' competitions at MIT, there were the cheers, the foot stomping, and hundreds of excited and very talented kids with some amazing robots. In this case, the Portland Rose Quarter was packed with grandstands full of supporters and a separate floor for the various team's 'pits' that exhibited the same energy and excitement as the competition arena. As someone I have long admired, inventor Dean Kamen's vision for FIRST was clearly evident in the several thousand

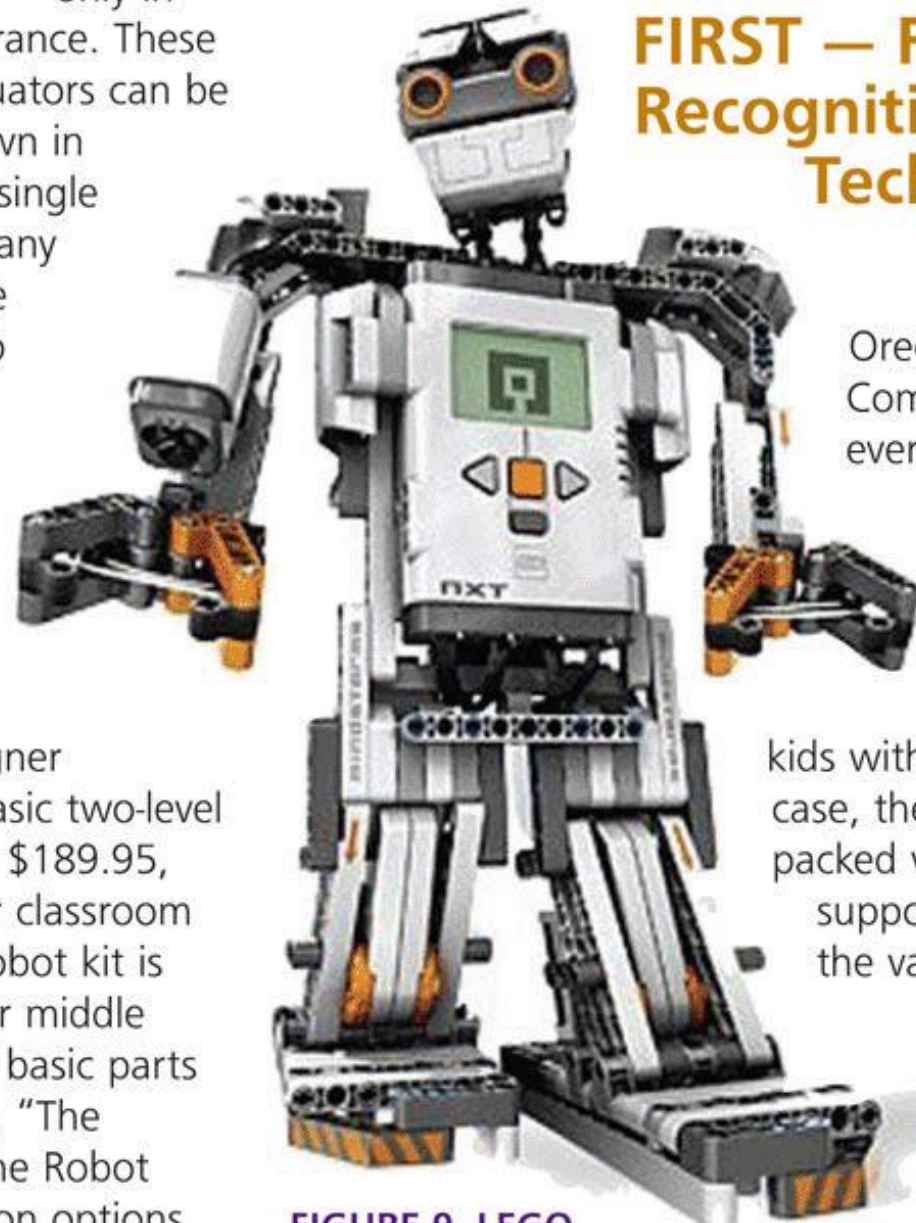
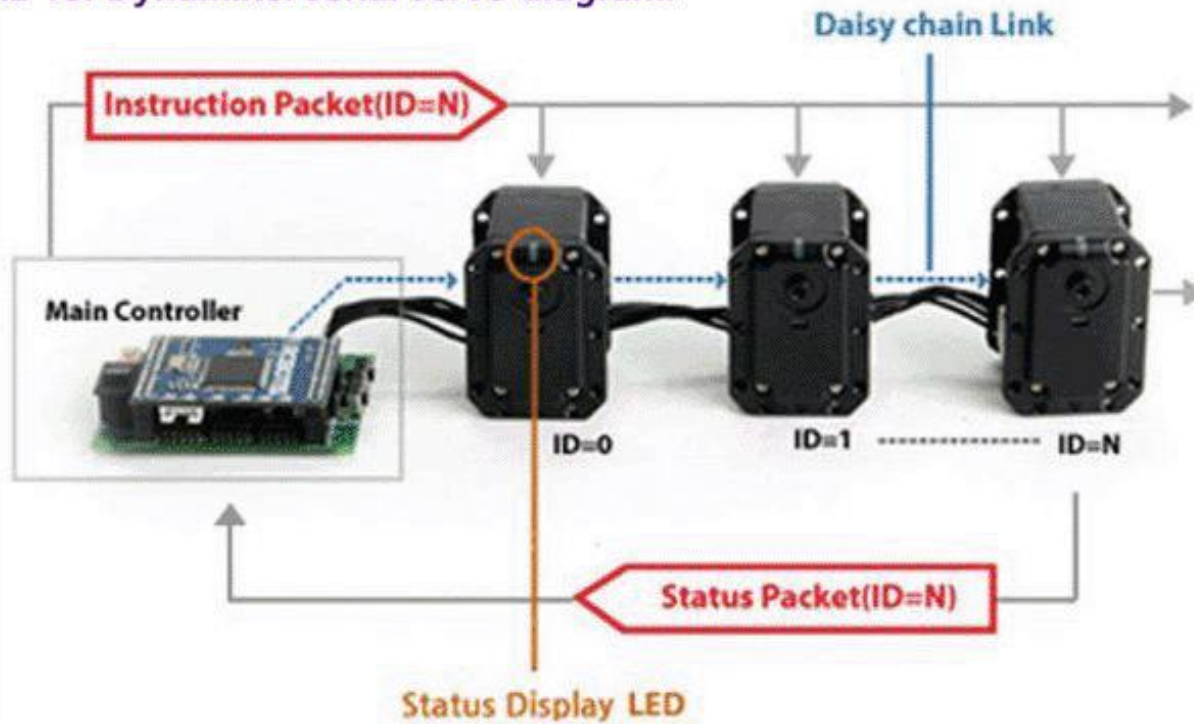


FIGURE 9. LEGO
Mindstorms NXT
2.0 robot.

FIGURE 10. Dynamixel serial servo diagram.



students, mentors, advisors, sponsors, and spectators. Woodie Flowers' spirit was also there, as he was the co-founder of the FIRST idea with Dean. I met many mothers who gladly called themselves 'robo moms' instead of 'soccer moms.' There were cheerleaders encouraging each team before their competition; flags were waving, banners were held high, and crowds clapped loudly as their team's results appeared on the scoreboard.

I won't go into detail about the individual robot's construction, the contest's rules or scoring, but I'll mention that every year's event is different. There was an autonomous period in the "logo motion" competition where the robots hung yellow 'ubertubes' on hooks for initial scores, then later there was scoring for hanging red, white, and blue FIRST image tubes by remote control. Finally, there was the release of tiny minibots from each large robot to race up pipes in the

arena. FIRST is the largest robot competition in the nation, and is open to all high school age students. **Figure 12** shows the FIRST Team #847 PHRED in their pit making final preparations for competition. Typical of many of the teams, Team #847 is composed of 26 students that include 10 girls. There are 10 mentors for the team and 20 sponsors — that's 56 people working hard and having a lot of fun teaching and learning. **Figure 13** is a scene from the competition in Portland.

Closing Thoughts

In closing, I would like to emphasize that it is volunteerism that makes contests such as FIRST, BEST, and so many equally great robot contests to be thriving beds of inspiration for kids across the country.

Sponsors range from JC Penny to Boeing to Autodesk and Mentor Graphics, as this takes a large financial worry off of budding teams. Education in robotics takes more than teachers. It takes funding of schools for the specialized materials needed for the classes. It takes parents who care and volunteer. It takes members of robotics groups such as the Seattle Robotics Society, who — along with a friend of mine, Kevin Ross — sent a team of volunteers down to Portland for this event.

Be a mentor, a judge in a contest, a supplier of materials for a needy school's classes, or, most of all, be a robotics education supporter. If you're interested in robotics engineering yourself, use a search engine to discover the many colleges that teach variations of science or join a FIRST team in your area. Learning about robotics

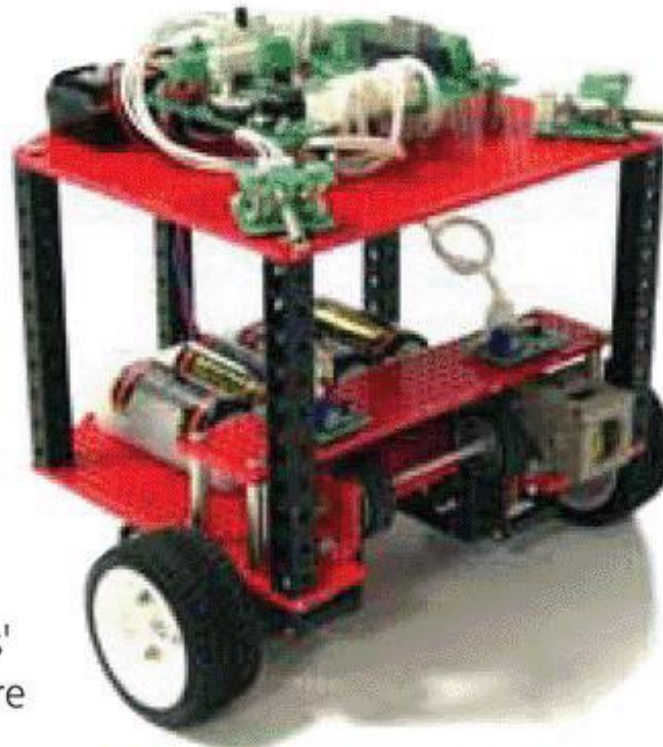


FIGURE 11. Robotech robot.



FIGURE 12. Team #847 PHRED gets ready in the pits.



FIGURE 13. Portland FIRST 2011 competition.

ROBO-LINKS

GPS MADE SIMPLE™



Linx Technologies.com

LOCATE • TRACK • MAP • FIND • NAVIGATE

THE ORIGINAL SINCE 1994

PCB-POOL®

Beta LAYOUT

- Low Cost PCB prototypes
- Free laser SMT stencil with all Proto orders

WWW.PCB-POOL.COM

RobotShop.com



Pololu

Robotics & Electronics

WWW.POLOLU.COM



ALL ELECTRONICS CORPORATION

Electronic Parts & Supplies Since 1967



AndyMark

Inspiring Mobility

www.andymark.com



superbrightleds.com

Component LEDs - LED Bulbs - LED Products

St. Louis, Missouri - USA Fast Online Ordering superbrightleds.com



For the finest in robots, parts, and services, go to www.servomagazine.com and click on **Robo-Links**.

Ultrasonic Ranging is EZ

- High acoustic power, auto calibration, & auto noise handing makes your job easy.
- Robust, compact, industrial (IP67) versions are available.

www.maxbotix.com



The 32-Bit Micro Experimenter Board

Now available in the **Nuts & Volts** webstore.

\$93.95 Subscriber Discount Available!

Includes Free Software!

For more information and kits, please visit: www.nutsvolts.com or call 800 783-4624



INVEST in your BOT!



HITEC

12115 Paine Street • Poway, CA 92064 • 858-748-6948 • www.hitecusa.com

To Advertise: Call 951-371-8497

ADVERTISER INDEX

All Electronics Corp.	21, 81	Hitecnic	66	Robot Power	35
AndyMark	16, 81	I²I Controls	41	Servo City/Robot Zone	83
AP Circuits	35	Linx Technologies	81	Solarbotics/HVW	7
AUVSI	82	Lynxmotion, Inc.	17	Sunstone Circuits	Back Cover
BaneBots	7	Maxbotix	81	superbrightleds.com	81
Blue Wolf	9	Minds-i	41	Technological Arts	21
Cross The Road Electronics	75	PCB Pool	59, 81	The Robot Marketplace	35
Firgelli	21	Polaris	13	Vantec	35
GSS Tech Ed	48	Pololu Robotics & Electronics	3, 81	Weirdstuff Warehouse	21
HiTec	2, 81	RobotShop, Inc	19, 81		

*Where the World of Robotics &
Unmanned Systems Comes together*

**AUVSI's
Unmanned Systems
North America 2011
16-19 AUGUST
WASHINGTON DC USA**



SAVE *the* **DATE**

Over 450 Exhibits

Over 50 Countries

Over 6,000 Attendees

- 4 Days of Education and Exhibits!
- LIVE Demonstration Areas for Air, Ground and Maritime Vehicles
- Broad Industry Representation in Civil, Commercial, and Government Markets
- The World's Largest Gathering for the Unmanned Systems and Robotics Communities

 **AUVSI**
ASSOCIATION FOR UNMANNED
VEHICLE SYSTEMS INTERNATIONAL

SERVOCITY

A DIVISION OF ROBOTZONE, LLC.

WE HAVE THE PARTS FOR YOUR IDEAS!

Specializing in Servos, Radios, Motors, Pan & Tilt Systems, Batteries, Wiring, Connectors, Gearboxes, and much more!

HITEC

FREE SERVO WHEELS!

(2) Hitec 1.875" Direct Mount Servo Wheels - See Below for Details.



Radio Systems



Hitec Servos



Hubs & Adaptors



Batteries & Chargers



Servo Arms



Joysticks



Servo Linkages



Linear Servos



Wheels & Tires



Gearmotors



Hardware



Pan & Tilts



Speed Controls



Bearings & Bushings



8 - Servo Controller



Dual Servo Driver



Shafts & Tubing



Power Connectors



Actuators



Servo Extensions



Couplers & U-Joints



Wiring



Gears & Sprockets



4 - Servo Recorder



Servo Power Gearboxes

How do I get my FREE Hitec Servo Wheels?

Enter promotional code **0611SW** in the "SCMPC#" field at the time of checkout and we will include a free pair of 1.875" Hitec Servo Wheels with your purchase. Limit one pair per customer, per order.

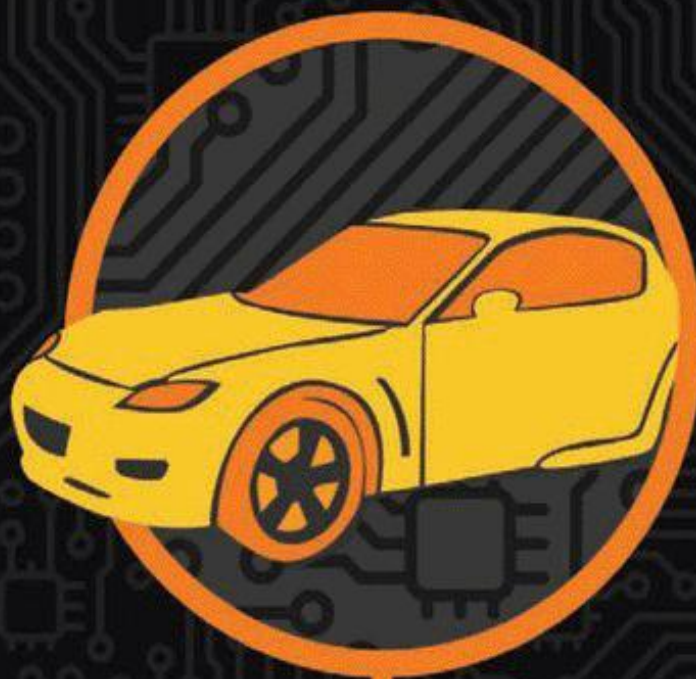
Offer valid while supplies last or thru June 30, 2011.

To view our entire selection of products, visit us online at

WWW.SERVOCITY.COM

Copyright 1999-2011 Robotzone, LLC. - All Rights Reserved
ServoCity is a registered trademark of Robotzone, LLC.

Phone: (620) 221-0123
E-mail: sales@servocity.com



NO MATTER WHAT THE IDEA YOUR PCB PROTOTYPES SHOULD BE THE EASY PART

QUOTE & ORDER PCBs ONLINE AT WWW.SUNSTONE.COM OR CALL 1-800-228-8198



THE EASIEST PCB COMPANY TO DO BUSINESS WITH



ValueProto™



PCBexpress®



Full Feature

Sunstone Circuits® pioneered the online ordering of printed circuit boards and is the leading PCB solutions provider with more than 35 years of experience in delivering quality prototypes and engineering software. With this knowledge and experience, Sunstone is dedicated to improving the PCB prototyping process from quote to delivery (Q2D®).

Did You Know? Sunstone Offers:

- Controlled impedance testing
- Fine lines and spacing [.003]
- RoHS compliant finishes
- Free 25-point design review
- Free shipping & no NRE's
- Flex / Rigid Flex Boards
- Online Quote & Order
- PCB123® design software
- RF / Exotic Materials
- Over 99% on-time or early delivery
- Best PCBs in the industry
- Live customer support 24/7/365